

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-181676  
 (43)Date of publication of application : 23.07.1993

(51)Int.Cl. G06F 9/38  
 G06F 9/38  
 G06F 9/38  
 G06F 11/00  
 G06F 15/16

(21)Application number : 04-082490 (71)Applicant : TOSHIBA CORP  
 (22)Date of filing : 03.04.1992 (72)Inventor : AIKAWA TAKESHI  
 MINAGAWA KENJI  
 SAITO MITSUO  
 TAKEDA JOJI

(30)Priority

Priority number : 03 73364 Priority date : 05.04.1991 Priority country : JP

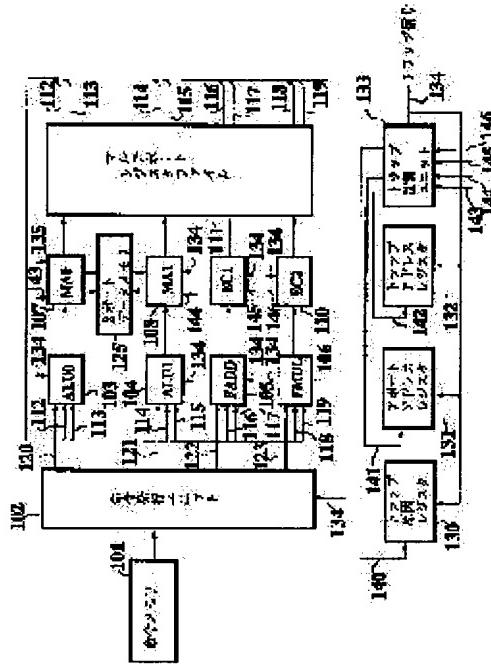
**(54) PARALLEL PROCESSING TYPE PROCESSOR SYSTEM**

**(57)Abstract:**

**PURPOSE:** To provide the parallel processing type processor system such as a super scalar processor, etc., integrating a trap, which can be processed without prolonging cycle time, and a stall control function.

**CONSTITUTION:** This system is provided with N (N is an integer) computing elements 103-106 to execute plural instructions at the same time, instruction supplying means 102 for supplying the instructions to be executed by the N computing elements 103-106, and trap control means for controlling the N computing elements 103-106 so as to abort all the processing of M ( $M \geq N$ , M is an integer) instructions simultaneously supplied to the N computing elements 103-106 when a processing exception is generated in the case of executing any one of these M instructions while supplying the M instructions from the instruction supplying means 102 to the N computing elements 103-106 at the same time.

Thus, the control of the trap and stall can be more efficiently executed without prolonging the cycle time and lowering the clock frequency of the system.



(19)日本国特許庁 (JP)

## (12) 公開特許公報 (A)

(11)特許出願公開番号

特開平5-181676

(43)公開日 平成5年(1993)7月23日

(51)Int.Cl. <sup>5</sup>	識別記号	序内整理番号	F I	技術表示箇所
G 0 6 F 9/38	3 8 0 B	9290-5B		
	3 1 0 X	9290-5B		
	3 7 0 X	9290-5B		
11/00	3 1 0 G	7313-5B		
15/16	3 9 0 Z	9190-5L		

審査請求 未請求 請求項の数2(全26頁)

(21)出願番号	特願平4-82490	(71)出願人	000003078 株式会社東芝 神奈川県川崎市幸区堀川町72番地
(22)出願日	平成4年(1992)4月3日	(72)発明者	相川 健 神奈川県川崎市幸区小向東芝町1 株式会社東芝総合研究所内
(31)優先権主張番号	特願平3-73364	(72)発明者	皆川 健二 神奈川県川崎市幸区小向東芝町1 株式会社東芝総合研究所内
(32)優先日	平3(1991)4月5日	(72)発明者	斎藤 光男 神奈川県川崎市幸区小向東芝町1 株式会社東芝総合研究所内
(33)優先権主張国	日本(JP)	(74)代理人	弁理士 三好 秀和 (外1名) 最終頁に続く

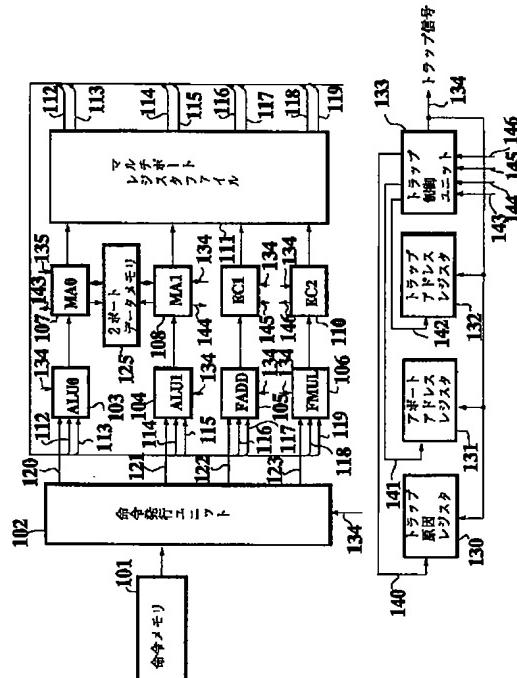
(54)【発明の名称】並列処理型プロセッサシステム

## (57)【要約】

【目的】サイクル時間を増加することなく処理可能なトラップとストールの制御機能を組み込んだスーパースカラプロセッサ等の並列処理型プロセッサシステムを提供すること。

【構成】複数命令を同時実行するN個(Nは整数)の演算器と、前記N個の演算器により実行される命令を供給する命令供給手段と、M個(N≥M、Mは整数)の命令が前記命令供給手段から前記N個の演算器に同時に供給され、これらM個の命令中の少なくとも一つの命令実行において処理例外が発生したとき、前記N個の演算器に同時に供給された前記M個の命令の処理を全てアボートするように前記N個の演算器を制御するトラップ制御手段と、を備える並列処理型プロセッサシステム。

【効果】サイクル時間を増加させることなく、スーパースカラプロセッサ等の並列処理型プロセッサシステムにおいてシステムのクロック周波数を低下させずにトラップとストールの制御をより効率的に行うことが可能となる。



## 【特許請求の範囲】

【請求項1】 複数命令を同時実行するN個（Nは整数）の演算器と、前記N個の演算器により実行される命令を供給する命令供給手段と、M個（N≥M、Mは整数）の命令が前記命令供給手段から前記N個の演算器に同時に供給され、これらM個の命令中の少なくとも一つの命令実行において処理例外が発生したとき、前記N個の演算器に同時に供給された前記M個の命令の処理を全てアボートするように前記N個の演算器を制御するトラップ制御手段と、を備えたことを特徴とする並列処理型プロセッサシステム。

【請求項2】 N個の演算器にM個（N≥M）の命令を同時に供給するステップと、これらM個の命令中の少なくとも一つの命令実行において処理例外が発生したとき、前記N個の演算器に同時に供給された前記M個の命令の処理を全てアボートするように前記N個の演算器を制御するステップと、を備えたことを特徴とする並列処理型プロセッサシステムの制御方法。

## 【発明の詳細な説明】

## 【0001】

【産業上の利用分野】 本発明は、命令レベルでの並列実行を行うための並列処理型プロセッサシステムに関し、より詳細には、並列処理型プロセッサシステムにおけるトラップ処理とストール処理の制御機能に関するものである。

## 【0002】

【従来の技術】 近年、高速パーソナルコンピュータに対して高まりつつある要望に応るものとして、機械語命令レベルで並列処理が可能なスーパースカラプロセッサあるいはV L I Wと呼ばれるC P Uが開発されV L S Iチップとして実現されている。この様な並列処理C P Uにおいては、R I S Cの命令を基本的な命令セットとして用い、複数命令を同時にフェッチ、実行することにより、処理性能を向上させている。特に、スーパースカラプロセッサは、従来の命令レベルで逐次処理を行うR I S Cを実現しユーザプログラムレベルで互換性を保つことが可能なようなアーキテクチャを有し、計算機ユーザからの期待が高い。

【0003】 このような従来のトラップ制御方法を採用する命令レベルで並列処理を行うことができるプロセッサシステムは、図19に示すような構成を有している。

【0004】 この図19の構成は、命令レベルで並列処理を実行可能なプロセッサシステムを実現するものであり、Fステージ（フェッチ）、Dステージ（デコード）、Eステージ（実行）、Mステージ（メモリアクセス）及びWステージ（レジスタライトバック）の5段のパイプラインステージを有し、各命令の長さが1ワード

長（32ビット）である。

【0005】 図19に示されるように、このプロセッサシステムは、命令を格納するための命令メモリ1と；Fステージで4ワードバウンダリの4つの命令を命令メモリ1から同時にフェッチし、Dステージで4つのフェッチされた命令間のデータ依存関係及び制御依存関係を考慮し、Eステージで命令供給線20、21、22及び23を介して実行可能な命令を供給する命令発行ユニット2と；命令供給線20及び21から供給される命令に従って、Eステージで算術論理演算及びメモリアドレス計算を実行する算術論理演算ユニット（ALU0及びALU1）3及び4と；命令供給線22から供給される命令に従ってEステージで浮動小数点加減算を行う浮動小数点加算器（FADD）5と；命令供給線23から供給される命令に従ってEステージで浮動小数点乗除算を行う浮動小数点乗算器（FMUL）6と；ALU03及びALU14の出力に従って、Mステージで2ポートデータメモリ25に対してメモリアクセス処理を行うメモリアクセスユニット（MA0及びMA1）7及び8と；FADD5及びFMUL6の出力に従って、各々のMステージでの浮動小数点計算における例外チェックを行うための浮動小数点例外チェックユニット（EC1及びEC2）9及び10と；WステージでMA07、MA18、EC19及びEC210の出力を受ける4つの書き込みポートと、Eステージでオペランドデータ供給線12～19を介してALU03、ALU14、FADD5及びFMUL6へオペランドデータを供給するための8つの読み込みポートとからなる12のポートを有するマルチポートレジスタファイル11と、を備えている。

【0006】 この図19の構成においては、MA07及びMA18によってページフォールトやオーバーフローなどの算術演算例外トラップが発生し、又、EC19及びEC210によって浮動小数点演算例外トラップが発生する。

【0007】 このような例外トラップに対処するため、このプロセッサシステムには更に、トラップ発生の原因を格納するためのトラップ原因レジスタ30と、トラップ発生を引き起こした命令のアドレスを格納するためのトラップアドレスレジスタ32と、トラップ要求信号線43～46を介して送られる、MA07、MA18、EC19及びEC210からのトラップ原因を受け、それに応じてトラップ信号をトラップ信号線34～38にアサートし、信号線40及び42を介してトラップ原因レジスタ30及びトラップアドレスレジスタ32への入力を適宜発生させる、トラップ制御ユニット33と、を備えている。

【0008】 トラップ信号線34におけるトラップ信号は、命令発行ユニット2、ALU03及びALU14へ送られ、一方、トラップ信号線35～38におけるトラップ信号は、各々、MA07、MA18、EC19及びEC210へ送られる。トラップ制御ユニット33からのトラップ信号に応じて、実行無効化フラグが各部に立てられ、その後の

パイプラインステージでの命令の処理をアボートする一方、命令発行ユニット2は予め定められたトラップ処理ルーチンに関する命令フェッチを開始し、このトラップ処理ルーチンにおいて、トラップ原因レジスタ30及びトラップアドレスレジスタ32に格納されたトラップ原因及びトラップアドレスが使用される。

【0009】更に詳細には、トラップ制御ユニット33は図20に示されるような構成を有している。即ち、トラップ制御ユニット33は更に、Mステージで現在実行される命令のアドレスの下位2ビットを除くことによって得られるワードアドレスの共通部分を格納するMステージプログラムカウンタ(MPC)51と；MステージでMA0 7、MA1 8、EC1 9及びEC2 10によって現在実行される命令のアドレスの下位2ビットによって示されるワードアドレスの個別部分を格納するためのMステージサブプログラムカウンタ(submpc1, submpc2, submpc3及びsubmpc4)53、54、55及び56と；Mステージサブプログラムカウンタ53～56の中で最小のエントリを、MPC51のエントリと組み合わせてトラップアドレスレジスタ32に供給されるトラップアドレス42を発生するための出力47として出力し、最小のエントリを有するMステージサブプログラムカウンタ53～56に対応するトラップ要求信号線43～46のうちの1つを介して送られるトラップ原因をトラップ原因レジスタ30に供給されるトラップ原因40として出力するトラップデータ生成ユニット57を備えている。

【0010】ここで、トラップ信号線34におけるトラップ信号はトラップ要求信号線43～46のいずれか1つからトラップ原因を受けた時にアサートされ、各トラップ信号35～38は、トラップ要求信号線43～46の1つからトラップ要求を受け、対応するMステージサブプログラムカウンタ53～56の1つが、トラップ要求を発生したMステージサブプログラムカウンタ53～56のうちの1つにおけるエントリより大きいか又は等しいエントリを有する時にアサートされる。

【0011】図21は図19のプロセッサシステムによって実行されるプログラムの一例を示す図であり、図22は、上述の従来のトラップ制御方法を用いた図19のプロセッサシステムによるパイプライン処理において、図21のプログラムが実行された時に「load」命令でページフォールトが起こった場合の進行状況を示すものであり、斜線部分はアボートされた命令を示す。図22に示すように、従来のトラップ制御においては、プログラム実行の進行中にn+2番目の「load」命令の実行によりトラップが発生したとき、命令番号がn+2より大きいか又は等しい命令だけがアボートされる。

### 【0012】

【発明が解決しようとする課題】しかしながら、このような従来のトラップ制御方法においては、C+3サイクル目にMA1 8によってページフォールトが検出されてト

ラップ要求がトラップ要求線44を介して示されたときに、トラップ信号線35～38におけるトラップ信号はMステージサブプログラムカウンタ53～56におけるエントリが互いに比較されてどれがどれより大きいかが決定されるまで決められず、そのような比較処理を施すサイクル時間がかなり長くなるためにクロック周波数の低下を引き起こすという問題があった。

【0013】又、RISCは、単純なデータパス及び簡易な制御回路を有する構成を必要としており、複数のR10 IISCデータパスを有するスーパースカラプロセッサのデータパスはさほど複雑ではないが、スーパースカラプロセッサの制御回路は、命令供給制御等を必要とするために非常に複雑となる。特に、OSのようなソフトウェアのサポートがないと処理の続行が不可能になる、いわゆる例外と呼ばれるケースの処理のためのハードウェアが非常に複雑になり、そのようなハードウェアの設計に非常に時間を要するために、このようなハードウェアがスーパースカラプロセッサの実現におけるクリチカルパスとなる場合が多いという問題があった。

【0014】本発明は、この様な従来技術の課題を解決するためになされたもので、サイクル時間を増加させることなく処理可能で、システムにおけるクロック周波数の低下を防げるようなトラップ及びストール制御機能を組み込んだ、スーパースカラプロセッサ等の並列処理型プロセッサシステムを提供することを目的とするものである。

### 【0015】

#### 【発明の構成】

### 【0016】

【課題を解決するための手段】上記課題を解決するため本発明の並列処理型プロセッサシステムは、複数命令を同時実行するN個(Nは整数)の演算器と、前記N個の演算器により実行される命令を供給する命令供給手段と、M個(N≥M、Mは整数)の命令が前記命令供給手段から前記N個の演算器に同時に供給され、これらM個の命令中の少なくとも一つの命令実行において処理例外が発生したとき、前記N個の演算器に同時に供給された前記M個の命令の処理を全てアボートするよう前記N個の演算器を制御するトラップ制御手段とを備えたことを特徴とする。

【0017】又、本発明の並列処理型プロセッサシステムの制御方法はN個の演算器にM個(N≥M)の命令を同時に供給するステップと、これらM個の命令中の少なくとも一つの命令実行において処理例外が発生したとき、前記N個の演算器に同時に供給された前記M個の命令の処理を全てアボートするよう前記N個の演算器を制御するステップとを備えたことを特徴とする。

### 【0018】

【作用】本発明においては、命令供給手段によりN個の演算器に同時に供給されたM個の命令の中の1つ以上の

命令実行過程で処理例外が発生したとき、同時に供給されたM個の命令の実行をすべて中止する手段を備えたので、同時に演算器に供給されたM個の命令の処理過程でどれか1つでも例外を発生したらM個の命令をすべてアポートしてOS内の処理ルーチンへディスパッチすることにより、例外処理のためのハードウェアが簡単化される。

#### 【0019】

【実施例】以下、図1から図3を参照して、本発明に係る並列処理型プロセッサシステムの第1の実施例を詳細に説明する。

【0020】この図1の構成は、命令レベルで並列処理を行うことができるプロセッサシステムを実現するものであり、Fステージ(フェッチ)、Dステージ(デコード)、Eステージ(実行)、Mステージ(メモリアクセス)及びWステージ(レジスタライトバック)の5段のパイプラインステージがあり、各命令の長さは1ワード長(32ビット)である。

【0021】この第1実施例においては、図1に示すように、プロセッサシステムは、命令を格納するための命令メモリ101と；Fステージで4ワードバウンダリの4つの命令を命令メモリ101から同時にフェッチし、Dステージで4つのフェッチされた命令間のデータ依存関係及び制御依存関係を考慮し、Eステージで命令供給線120、121、122及び123を介して実行可能な命令を供給するためのする命令発行ユニット102と；命令供給線120及び121から供給される命令に従って、Eステージで算術論理演算及びメモリアドレス計算を実行する算術論理演算ユニット(ALU0及びALU1)103及び104と；命令供給線122から供給される命令に従ってEステージで浮動小数点加減算を行う浮動小数点加算器(FADD)105と；命令供給線123から供給される命令に従ってEステージで浮動小数点乗除算を行う浮動小数点乗算器(FMUL)106と；ALU0103及びALU1104の出力に従って、Mステージで2ポートデータメモリ125に対するメモリアクセス処理を行うメモリアクセスユニット(MA0及びMA1)107及び108と；FADD105及びFMUL106の出力に従って、各々のMステージでの浮動小数点計算における例外チェックを行うための浮動小数点例外チェックユニット(EC1及びEC2)109及び110と；WステージでMA0107、MA1108、EC1109及びEC2110の出力を受ける4つの書き込みポートと、Eステージでオペランドデータ供給線112～119を介してALU0103、ALU1104、FADD105及びFMUL106へオペランドデータを供給するための8つの読み込みポートとかなる12のポートを有するマルチポートレジスタファイル111と、を備えている。

【0022】この図1の構成においては、MA0107及びMA1108によってページフォールトやオーバーフロー

などの算術演算例外トラップが発生し、又、EC1109及びEC2110によって浮動小数点演算例外トラップが発生する。

【0023】このような例外トラップに対処するため、このプロセッサシステムは更に、トラップ発生の原因を格納するためのトラップ原因レジスタ130と；トラップによって実行が遮断されている命令中で最も小さいアドレスを有する命令のアドレスを格納するためのアポートアドレスレジスタ131と；トラップ発生を起こした命令のアドレスを格納するためのトラップアドレスレジスタ132と；トラップ要求信号線143～146を介して送られる、MA0107、MA1108、EC1109及びEC2110からのトラップ原因を受け、それに応じてトラップ信号をトラップ信号線134を介してアサートし、一方、信号線140、141及び142を介してトラップ原因レジスタ130、アポートアドレスレジスタ131、トラップアドレスレジスタ132への入力を適宜発生させるトラップ制御ユニット133と、を備えている。

【0024】トラップ信号線134におけるトラップ信号は、命令発行ユニット102、ALU0103、ALU1104、FADD105、FMUL106、MA0107、MA1108、EC1109及びEC2110へ各々送られる。トラップ制御ユニット133からのトラップ信号に応じて、実行無効化フラグが各部に立てられ、その後のパイプラインステージで命令の処理をアポートする一方、命令発行ユニット102は予め定められたトラップ処理ルーチンに関する命令フェッチを開始し、このトラップ処理ルーチンにおいて、トラップ原因レジスタ130、アポートアドレスレジスタ131及びトラップアドレスレジスタ132に各々格納されているトラップ原因、アポートアドレス及びトラップアドレスが使用される。

【0025】更に詳細には、トラップ制御ユニット133は図2に示されるような構成を有している。即ち、トラップ制御ユニット133は更に、Mステージで現在実行される命令のアドレスの下位2ビットを除くことによって得られるワードアドレスの共通部分を格納するMステージプログラムカウンタ1(MPC)151と；Mステージで現在実行される命令のうち最も小さいアドレスを有する命令のアドレスの下位2ビットによって示されるものであってMPC151のエントリと組み合わされてアポートアドレスレジスタ131に供給されるアポートアドレスを発生することになるワードアドレスの個別部分を格納するMステージプログラムカウンタ2(mpcc)152と；MステージでMA0107、MA1108、EC1109及びEC2110によって現在実行される命令のアドレスの下位2ビットによって示されるワードアドレスの個別部分を格納するためのMステージサブプログラムカウンタ(submpc1, submpc2, submpc3及びsubmpc4)153、154、155及び156と；Mステージサブ

ログラムカウンタ153～156の中で最小のエントリを、MPC151のエントリと組み合わされてトラップアドレスレジスタ132に供給されるトラップアドレス142を発生するための出力147として出力し、最小のエントリを有するMステージサブプログラムカウンタ153～156に対応するトラップ要求信号線143～146のうちの1つを介して送られるトラップ原因をトラップ原因レジスタ130に供給されるトラップ原因140として出力するトラップデータ生成ユニット157と；トラップ原因をトラップ要求信号線143～146のいずれか1つから受けたときにアサートされるトラップ信号をトラップ信号線134に対して発生するトラップ信号生成ユニット158と、を備えている。

【0026】図3は、図1のプロセッサシステムにおけるパイプライン処理において、前記図21のプログラムが実行された時に「load」命令でページフォールトが起こった場合の進行状況をしめすものであり、斜線部分はアボートされた命令を示す。図3に示すように、図1の第1実施例においては、プログラム実行の進行中にn+2番目の「load」命令の実行によりトラップが発生したとき、トラップを引き起こしているn+2番目の命令と同時にフェッチされた命令番号がnからn+3の命令全てがアボートされる。図22に示される従来の場合と比較すると、アボートされる命令の数が従来の場合の方が少ないので従来の場合の方が効率的であると思われるかも知れないが、既に述べたように従来の場合は適切なトラップ信号を決定する時間が必要なために、従来の場合の方がこの第1実施例よりサイクル時間が長くなり、実際にはこの第1実施例の方が効率が高くなる。

【0027】より詳しく説明すると、図3に示されるパイプライン処理において、n番目、n+1番目、n+2番目及びn+3番目の命令全てがサイクルC+3でMステージに入り、n番目の「fadd」処理についてのMステージ処理がEC1109で行われ、n+1番目の「add」処理についてのMステージ処理がMAO107で行われ、n+2番目の「load」処理についてのMステージ処理がMA1108で行われ、n+3番目の「fmul」処理についてのMステージ処理がEC2110で行われる。このサイクルC+3において、ページフォールトがMA1108で検出されたとき、MA1108はトラップ要求信号線144を介してトラップ制御ユニット133にページフォールトラップの発生を知らせる。ここで、このサイクルC+3において、MPC151は、Mステージで現在実行されているn番目～n+3番目の命令の下位2ビットを除いたワードアドレスを格納し、mpc152は、Mステージで現在実行されている命令中最小のアドレスを有するn番目の命令の下位2ビットを示すワードアドレスを格納しており、この場合これは0である。一方、submpc1153、submpc2154、submpc3155及びsubmpc4156は、MA0107、MA1108、EC1109及びEC2110の各

々で現在実行されているn+1番目、n+2番目、n番目及びn+3番目の命令の下位2ビットを各々示すワードアドレスを格納している。

【0028】トラップ信号生成ユニット158は、トラップ要求信号線143～146のいずれかにトラップ要求があるとき、トラップ信号線134のトラップ信号をアサートする。トラップ信号線134におけるアサートされたトラップ信号は、命令発行ユニット102、ALU0103、ALU1104、MA0107、MA1108、FADD105、FMUL106、EC1109及びEC2110に供給されるので、これらのユニットにおける処理がアボートされると同時に、命令発行ユニット102は予め定められたトラップ処理ルーチンに関する命令フェッチを始める。

【0029】ここで、トラップデータ生成ユニット157がトラップ要求信号線144にトラップ要求を検出すると、このトラップ要求信号線144に対応するsubmpc2154のエントリが、信号線147に出力され、それがMPC151のエントリと組み合わされてトラップアドレス142、この場合n+2番目の命令のアドレス、を生成し、このn+2番目の命令のアドレスはトラップアドレス信号線142を介してトラップアドレスレジスタ132に格納される。

【0030】他方、MPC151のエントリをmpc152のエントリと組み合わせることによりアボートアドレス141が得られ、この場合のアボートアドレスであるn番目の命令のアドレスはアボートアドレス信号線141を介してアボートアドレスレジスタ131に格納される。

【0031】更に、トラップ要求を現在出しているトラップ要求信号線に付随するMステージプログラムカウンタ中で最小のアドレスを格納するsubmpc1153、submpc2154、submpc3155及びsubmpc4156の一つに対応するトラップ要求信号線143～146の1つにおける信号がトラップ原因信号線140へ出力され、トラップ原因レジスタ130に格納される。この場合、トラップ要求は、トラップ要求信号線144のみから出されているので、トラップ要求信号線144の信号がトラップ原因信号線140へ出力されトラップ原因レジスタ130に格納されることになる。

【0032】ここでは、n番目からn+3番目の命令全てが同時にフェッチされる場合を説明しているが、4つの命令間にデータの依存性があるためにmpcのエントリが0でない場合がありうる。又、2個以上のトラップ要求がトラップ要求信号線143～146において検出された場合には、submpc1153、submpc2154、submpc3155及びsubmpc4156の内でMステージで現在実行される命令中最小のアドレスを格納するもののエントリが信号線147に出力される。

【0033】この実施例においては、トラップを引き起

こす命令とアボートされる命令は異なるので、アボートアドレスレジスタ131とトラップアドレスレジスタ132の両方が必要である。

【0034】又、この実施例においては、トラップ要求信号線143～146のいずれであっても少なくとも1つトラップ要求があればすぐにトラップ信号はアサートされるのでトラップ信号線におけるエントリは非常に速く決めることができる。他方、トラップ原因レジスタ130に格納されるトラップ原因140及びトラップアドレスレジスタ132に格納されるトラップアドレス142は、Mステージサブプログラムカウンタ153～156におけるエントリを比較した結果として決定されるため、これらはずっと後にならないと決められない。しかし、一般的に言って、メモリアクセスをアボートするためにトラップ信号線におけるエントリが速く決められる必要はあるが、レジスタに格納されるトラップデータはずっと遅く決められても構わないので、この実施例においてはプロセッサシステムのサイクル時間を長くする必要は生じない。

【0035】従って、この第1実施例によると、システムにおけるクロック周波数の低下を防げるよう、サイクル時間を増加させることなく機能できるトラップ制御機能を組み込んだスーパースカラプロセッサ等の並列処理型プロセッサシステムを提供することが可能となる。

【0036】次に、図4から図6を参照して、本発明に係る並列処理型プロセッサシステムの第2の実施例について詳細に説明する。

【0037】この第2実施例は、上述の第1実施例の応用であり、以下の理由から、第1実施例のトラップ制御機能に加え、更に、ストール制御機能を組み込んだものである。

【0038】即ち、第1実施例のシステムを変形して、図4に示すように、システムが更に、バス線を介して命令キヤッシュメモリ101A及び2ポートデータキヤッシュメモリ125Aに接続されるメインメモリ161とI/O装置162などを含むようにした場合、トラップ制御機能だけではトラブルが起る場合がある。トラブルの原因は、ここでは、メモリアドレスにマップされる複数のI/Oレジスタを通常含んでいるI/O装置162である。このようなI/Oレジスタへのアクセスは、I/Oレジスタのアドレスを特定するメモリアクセス命令と同一の命令によってなされ、キヤッシュは通常この目的には使用されない。I/Oレジスタは、コマンド及びパラメータをI/O装置162に設定するためとI/Oレジスタのステータスをプロセッサ側に示すために用いられ、例えばI/O装置162のステータスを読み込むためのアクセスに応じて、プロセッサ側でI/O装置162のステータスを読み込んだときにステータスレジスタを消去するような形で、I/Oレジスタの内部状態を変えてしまうタイプのI/Oレジスタがある。

【0039】そのようなI/O装置162を使用する場合、上述の第1実施例のトラップ制御機能のみを有するシステムでは、以下のような問題に直面することになる。即ち、このシステムにおいては、2つのメモリアクセス命令が同時実行可能なような2ポートデータキヤッシュメモリ125Aについて、2つのアクセスユニット107及び108が設けられている。従って、メモリアクセスユニット107及び108において、2つのI/Oアクセス命令がMステージに同時に到達する場合がある。しかし、I/Oアクセスの目的のためのラインはバス線1601のみしかないので、2つのI/Oアクセス命令を同時に処理することは不可能である。この結果、2つのI/Oアクセス命令のうちの最初の1つの処理については2番目のI/Oアクセス命令の処理が始まる前にI/O装置162の側で完了していかなければならない。

【0040】このような状況において、2番目のI/Oアクセス命令の処理でバスエラーなどの例外が起きる可能性がある。そのような場合、もしシステムが上記第1実施例のトラップ制御機能しか用いていないと、両方のI/Oアクセス命令がアボートされることになる。従って、適切なトラップ処理ルーチンを行った後に元のプログラムを再度実行するとき、I/O装置162の側においては最初のI/Oアクセス命令は終わったと見なされているにも関わらず、これがもう一度実行されることになる。この場合、最初のI/Oアクセス命令が偶然に、ステータスレジスタを読む命令であると、ステータスレジスタはトラップ発生の前に既に一度読まれているので、このステータスレジスタの内容は既に消去されており、最初のI/Oアクセス命令を再度実行するときにはもはや正しいものではなくくなってしまっているためトラブルが生じる。

【0041】上記のトラブルは基本的に、Dステージでは検出できない同一演算リソースの使用に関する要求の競合がその後のパイプラインステージで起こり得ることによる。それ故、もしDステージでこれらの2つのメモリアクセス命令が実際に2つのI/Oアクセス命令であることを検出できれば、これらの2つのメモリアクセス命令をプロセッサに同時に供給しないことによってトラブルは避けることが出来るが、その反面、I/Oアクセス命令のためだけに特別な命令を使用することを要求することになる。

【0042】図4に示される第2実施例の構成においては、この問題を、以下のように、そのような特別なI/Oアクセス命令を使用せずに解決するようにしている。

【0043】まず第1に、この第2実施例においては、命令がALU0103及びALU1104へ同時に供給されることになるとき、最初に実行されるべき、より小さいアドレスを有する命令がALU0103へ供給されるように、命令発行ユニット102AがALU0103及びALU1104へ

の命令供給を制御する。従って、メモリアクセス命令がMA0 107及びMA1 108へ同時に届いたときには、MA0 107が常に先に実行されるべきメモリアクセス命令を有していることになる。

【0044】第2に、図4の構成は、ストール要求信号がMA0 107、MA1 108、EC1 109及びEC2 110からストール要求信号線170、71、72及び173を介して各々供給されるストール制御ユニット163を更に含んでおり、このストール制御ユニット163はstall1信号180、stall2信号181及びstallv1信号182を命令発行ユニット102A、ALU0103、ALU1104、FADD105、FMUL106、MA0 107、MA1 108、EC1 109、EC2 110及びトラップ制御ユニット133へ出力して、以下に記載するような適当なストール制御を行う。

【0045】stall1信号180は、MA0 107、MA1 108、EC1 109及びEC2 110のいずれかからのストール要求があるときアサートされ、stall2信号181は、MA0 107、MA1 108、EC1 109及びEC2 110のいずれかからのストール要求があるときアサートされ、一方stallv1信号182は、Mステージの処理がMA1 108で現在実行されている命令のアドレスの下位2ビットを示す。

【0046】これらのstall1、stall2及びstallv1信号180～182の値に従って、このシステムにおけるパイプライン処理は以下のように制御される。

【0047】(1) ALU0103及びMA0 107のパイプライン<103、107>:

(a) Mステージ及びWステージ: stall2信号181がニゲートであれば、stall1信号180に関係なくパイプライン処理が行われる。

【0048】(b) Eステージ: stall1信号180がニゲートであれば、パイプライン処理が行われる。

【0049】(2) ALU1104及びMA1 108のパイプライン<104、108>: stall1信号180がニゲートであれば、パイプライン処理が行われる。

【0050】(3) FADD105及びEC1 109のパイプライン<105、109>及びFMUL106及びEC2 110のパイプライン<106、110>:

(a) Mステージ及びWステージ: stall1信号180がニゲートであれば、stall1信号180に関係なくパイプライン処理が行われる。

【0051】パイプライン処理は又、stall1信号180がアサートでありstall2信号がニゲートであって、stallv1信号182が、EC1 109及びEC2 110の各々においてMステージの処理が現在行われている命令の下位2ビットを格納するsubmpc3155及びsubmpc4 156の各々より大きい値を示している場合に行われる。

【0052】(b) Eステージ: stall1信号180がニゲートであれば、パイプライン処理が行われる。

【0053】又、この第2実施例においては、mpc 152に格納される値はこれらstall1、stall2及びstallv1信号180～182の値に従って以下の様に決定される。即ち、stall1信号180がアサートされないとき、mpc 152には、Eステージで実行された命令中の最小アドレスがロードされ、stall1信号180及びstall2信号181の両方がアサートされるときは、mpc 152は前の値を維持するが、stall1信号180がアサートされるがstall2信号181信号はアサートされないときは、mpc 152にはstallv1信号182によって示される値がロードされる。

【0054】従って、この第2実施例では、MA1 108だけからストール要求があれば、パイプライン<103、107>の処理が完了し、現在処理されている命令のアドレスがMA1 108での命令のアドレスより小さいときのみ、パイプライン<105、109>と<106、110>の各々の処理が完了する。

【0055】この結果、第1及び第2 I/Oアクセス命令が同時にMA0 107及びMA1 108に各々届いたとき、MA1 108はストール要求をストール制御ユニット163に出力するが、MA0 107は、1クロックサイクル以内でI/Oアクセス処理を完了させるのが不可能でない限り、ストール要求を出力しない。そして、EC1 109及びEC2 110もストール要求を出力しない時、stall1信号180のみがアサートされ、stall2信号182はアサートされない。そのような場合、パイプライン<303、107>は処理が進められ、mpc 152には、stallv1信号182の値がロードされて、システムの状態が2番目のI/Oアクセス命令よりも前の命令が実行完了している状態となる。

【0056】それ故、2番目のI/Oアクセス命令について例外が発生したときでも、最初のI/Oアクセス命令は再度実行されない。他方、パイプライン<104、108>の処理と、2番目のI/Oアクセス命令より大きいアドレスを有する命令についてのパイプラインの処理は、stall1信号180がニゲートになる迄ストールされる。

【0057】1クロックサイクル以内にI/Oアクセス処理を完了させるのが不可能であるためにMA0 107がストール要求を出力するときには、最初のI/Oアクセス命令が完了してstall2信号181がニゲートになるまで、stall1信号及びstall2信号182がアサートされるので、全てのパイプラインはストールされる。

【0058】図6は、図4のプロセッサシステムにおけるパイプライン処理において、図5のプログラムを実行した時にn+2番目の「ロード」命令でバスエラーが起こった場合の進行状況を示すものである。図6に示されるように、この図4の第2実施例においては、ストール要求がサイクルC+3においてn+2番目の命令の実行によって発生するとき、EC1 109で行われるn番目の「fad

d」処理についてのMステージ処理及びMA1 107で行われるn+1番目の「add」処理についてのMステージ処理は、次のサイクルC+4で終了するが、MA1 108でのn+2番目の「load」操作についてのMステージ処理及びEC2 110でのn+3番目の「fmul」操作についてのMステージ処理及びは次のサイクルC+4でストールされ、その後のサイクルC+5迄終了しない。一方、n+2番目の命令より大きいアドレスをもった後続の命令も全てサイクルC+4でストールされる。

【0059】ここで、命令の実行中に例外が起こる可能性を否定できないために命令の処理がストールされた後で、命令の実行中に実際に例外が発生したときには、命令の処理はアボートされる。さもなくば、命令実行中に実際には例外が起らなかったので命令の処理が再開される。

【0060】従って、この第2実施例によると、システムにおけるクロック周波数の低下を防げるよう、サイクル時間を増加させることなく機能できるトラップとストールの制御機能を組み込んだスーパースカラプロセッサ等の並列処理型プロセッサシステムを提供することが可能となる。

【0061】次に図7を参照して、本発明に係る並列処理型プロセッサシステムの第3の実施例を詳細に説明する。

【0062】この第3実施例は、より一般的な設定における上記第2実施例のストール制御機能の一般化を行ったものである。

【0063】この第3実施例においては、図7に示されるように、プロセッサシステムは、命令を格納する命令キャッシュメモリ(I-cache)201と；FステージでI-cache 201から4ワードバウンダリの4つの命令を同時にフェッチし、Dステージで4つのフェッチされた命令間のデータ依存関係及び制御依存関係を考慮し、Eステージで命令供給線220、221、222及び223を介して実行可能な命令を供給する命令発行ユニット202と；命令供給線220及び221から供給される命令に従って、Eステージで算術論理演算及びメモリアドレス計算を実行する論理演算ユニット(ALU0及びALU1)203及び204と；ALU0 203及びALU1 204からのコマンドに従ってEステージで整数の乗除算を行うための整数乗除算器205と；ALU0 203及びALU1 204からのアクセスされるデータを格納するためのデータキャッシュメモリ(D-cache)206と；命令供給線222から供給される命令に従ってEステージで浮動小数点の加減算を行うための浮動小数点加算器(FADD)207と；命令供給線223から供給される命令に従ってEステージで浮動小数点の乗算を行うための浮動小数点乗算器(FMUL)208と；FMUL 208からのコマンドに従ってEステージで浮動小数点の除算を行うための浮動小数点除算器(FDIV)209と；命令発行ユニット202

で発行される命令を特定するための命令アドレス生成ユニット210と；後に説明するトラップ制御及びストール制御を行うための制御ユニット211と；ALU0 203、ALU1 204の出力を格納するための整数レジスタファイル212と；FADD 207及びFMUL 208の出力を格納するための浮動小数点レジスタファイル213とを備えている。

【0064】上述の図4の第2実施例と同様に、図7のプロセッサシステムは、更に、I-cache 201及びD-cache 206にキャッシュされるデータを格納するためのメインメモリ214と；I/Oレジスタを含むI/O装置215と；メインメモリ214及びI/O装置215をI-cache 201及びD-cacheに接続させるバス線216とを有している。

【0065】加えて、図7のプロセッサシステムには、更に、I-cache 201とバス線216との間に設けられるプレデコーダ217と；命令発行ユニット202に接続されているレジスタスコアボード回路218とが組み込まれている。これらについては以下に詳細に説明する。

【0066】まず、この第3実施例におけるトラップとストールの制御処理について説明する。

【0067】一般に、並列処理型プロセッサシステムは、マシン語命令の実行の時に同じリソースについての競合する要求が発生するのを回避する機構、及び、マシン語命令間の実行順序の正当性を保持する機構を有する必要がある。ここで、実行順序の正当性とは、データ依存関係及び制御依存関係のコンシスティンシを意味する。

【0068】データ依存関係のコンシスティンシを維持するためには、実行順序をD→S関係、S→D関係及びD→D関係の内の1つに維持する必要がある。ここで、D→S関係は、先に実行すべき命令の結果を格納するリソースが、後で実行すべき命令に用いられるソースデータが読みだされるリソースと同一である関係を示す。S→D関係は、先に実行すべき命令の結果を格納するリソースが、後で実行すべき命令の結果を格納するためのリソースと同一である関係を示す。D→D関係は、先に実行すべき命令に用いられるソースデータが読みだされるリソースが、後で実行すべき命令に用いられるソースデータが読みだされるリソースと同一であることを示す。この図7の第3実施例においては、データ格納リソースが、レジスタ及びメモリの2つの態様で存在し、両者間のデータ依存関係のコンシスティンシを保つ必要がある。

【0069】制御依存関係とは、先の分岐命令とその後の命令との間の関係である。従来のVLIW型プロセッサにおける分岐命令については、この制御依存関係のコンシスティンシがコンパイラによって保たれている。しかし、この第3実施例の並列処理型プロセッサシステムでは、ユーザープログラムのオブジェクトコンパチビリティをもたせる必要があるので、制御依存関係のコンシス

テンシの保持はハードウェアにおいて実現させる必要がある。

【0070】この第3実施例においては、プロセッサシステムは、4ワードバウンダリの4つの命令を同時にフェッチし、その4つの命令間でインオーダーな順序で同時に発行ができる命令を各演算ユニットに発行して、インオーダーな順序で命令の実行が完了する。

【0071】この第3実施例においては、同じリソースを使用する競合する要求が発生するのを回避しデータ依存関係及び制御依存関係のコンシステンシを保持するためのハードウェアは、2つの機構を含んでいる。第1の部分は、命令発行ユニット202によって実現される命令発行機構であり、第2の部分は、制御ユニット211によって実現されるストール機構である。

【0072】命令発行機構を実現するために、命令発行ユニット202には、同じリソースを使用する競合する要求の発生を検出するためのプレデコーダ217が備えられており、これはキャッシュへのリフィル時（又はキャッシュスルーリード時）に同時にフェッチされた4つの命令の使用リソースと、同時にフェッチされた命令のうちで最小のアドレスを有する命令について競合があるかどうかとのマークを付ける。命令発行ユニット202は又、D→S関係及びD→D関係を検出するためのレジスタスコアボード回路218も備えており、プレデコーダ217によってマークされる競合を回避しつつ、レジスタスコアボード回路218によって検出されるデータ依存関係及び制御依存関係においてD→S及びD→D関係を保つことにより、Dステージでの命令発行処理をインオーダーな順序で行うことができる。S→D関係については、ソースレジスタの読み出し後に、命令発行をインオーダーな順序で行いインオーダーな順序で実行が完了することにより保たれる。

【0073】更に、この第3実施例においては、2つのメモリアクセス命令を同時に実行することが可能であるために、メモリリソースについてのデータ依存関係と同様、メモリリソースの競合関係のコンシステンシも保つ必要がある。命令発行ユニット202はDステージより前で処理を行うので、命令発行ユニット202はこれらの関係がないと仮定して処理を行っており、これらの関係が存在するかどうかを正確に確認することができない。これらの関係の存在はMステージになるまで正確には決められず、関係の存在の可能性を決めるることは可能であっても、命令供給処理をそのような可能性に基づいて過剰に制御してしまうと、パフォーマンスが著しく制限されてしまうことになる。

【0074】ストール機構においては、実行完了はインオーダーな順序に保たれ、命令発行機構で既に考慮されたもの以外のケース及び命令発行処理が速すぎて命令フェッチ処理が追いつかなくなつた場合についての制御を

行う。

【0075】インオーダーな順序での実行完了においては、Dステージから同時発行された命令は同時に実行完了するという基本原理に基づいた命令の実行完了が達成される。但し、ALU1204によってMステージで実行されるメモリアクセス命令Xによるメモリアクセス処理が1サイクル時間内に終了できない時には、上記第2実施例と同様に、命令Xによるストール要求に関わらず、このメモリアクセス命令Xより小さいアドレスを有する命令を実行完了し、実行完了した命令の数だけmpcが更新される。この機能を実行完了ステージにおけるグループ機能と呼ぶ。この第3実施例においては、ALU0203及びALU1204で命令を同時に実行するとき、命令発行ユニット202は、ALU0203が常により小さいアドレスを有する命令を実行するような制御を行うので、インオーダーな実行完了が保証される。この命令発行ユニット202による制御によって、Dステージでは検出することができないメモリリソースの使用に関する競合する要求の発生によるデッドロックをMステージにおいて防ぐことができる。

【0076】この様な、ストール要求を発している命令より小さいアドレスを有する命令のみについてパイプライン処理を完了するための制御は、他の状況、例えば、ALU0203でのMステージ処理が1サイクルで終了できないような場合や、FADD207又はFMUL208のE2ステージで実行される命令についてトラップの可能性が否定できないためにストール要求がアサートされている場合等にも同様に適用することが可能である。

【0077】命令発行ユニット202は、メモリ（キャッシュメモリを含む）に関する限り、同じリソースの使用に関する競合する要求の発生あるいはデータ依存関係を考慮しない。この第3実施例のプロセッサシステムはRISCと同じように、いわゆる「load, store」アーキテクチャを採用するので、同じリソースの使用に関する競合する要求の発生の防止及びデータ依存関係の保持を確保するためには、「load」及び「store」命令のみを考慮すれば充分である。D-cache206にはALU0203及びALU1204用の専用ポートがあるので、ALU0203及びALU1204の1つだけがD-cache206にアクセスしている限りは、リソースの競合は起こらない。故に、「load」及び「store」命令は外部メモリに同時にアクセスする際、データ依存関係はこの2つの命令が同じアドレスにアクセスしようとしている場合にのみ存在することになる。

【0078】メモリリソース競合は実行完了ステージにおける命令グループ機能によって解決される。メモリについてのデータ依存関係には、「リードアフターライド」、「ライトアフターリード」及び「ライトアフターライト」があり、そのコンシステンシはキャッシュメモリの側で保たれる。

【0079】次に図8から図18を参照して、図7の第3実施例におけるストール制御を詳細に説明する。

【0080】この第3実施例においては、ストールが生じる場合は以下のように要約することができる。

#### 【0081】M0busy

M1busy

Imis (I-cache miss)

FRbusy (FPUFA レジスタ書き込み競合)

FAexch (FADD例外チェック)

FMexch (FMUL例外チェック)

FDexch (FMUL例外チェック)

Fstall (強制ストール)

これらの場合の各々に対応するストール要求信号がアサートされる条件は以下のようなものである。

【0082】M0busy、M1busy：これらのタイプのストール要求信号は、メモリアクセス処理（キャッシュ及びI/Oに関するアクセス処理を含む）がMステージで $t$ 番目のサイクルに行われ、このメモリアクセス処理がこの $t$ 番目のサイクル中に完了できない時にアサートされる。

【0083】Imis：このタイプのストール要求信号は、新しい命令フェッチ要求があるがその命令フェッチが不成功の時に、アサートされる。ストールがこのImisストール要求によって起こった場合の例を図8から図12に示す。ここでは、Imisストール要求信号は、fpcに新しい値がロードされ「fpcen」をアサートされてから1クロックサイクル後に命令が命令レジスタにフェッチされない時、実際にアサートされる。

【0084】図8は命令フェッチ時に命令キャッシュミスによるストールが発生した場合を示している。

【0085】図9は命令フェッチ時に命令キャッシュミスによるストールが発生した後、システム中の他の部分で別のストール要求が発生し、命令キャッシュミスによるストールが他の部分での別のストールより先に解消された場合を示している。

【0086】図10は命令キャッシュミスによるストールがシステムの他の部分での別のストールと同時に発生し、他の部分での別のストールの方が命令キャッシュミスによるストールよりも先に解消した場合を示している。

【0087】図11は命令フェッチ時の命令キャッシュミスによるストールがジャンプ命令のジャンプ先で発生した場合を示している。

【0088】図12は命令キャッシュミスによるストールに伴うキャッシュリファイル動作中にジャンプが起こった場合を示している。

【0089】FRbusy：このタイプのストール要求信号は、FDIV209がパイプラインのE2ステージにあってFMUL208もこのE2ステージにある時にアサートされ、FDIV以外のパイプライン処理を1サイクルの間ストールす

ることによって、FMUL208とFDIV209間での浮動小数点レジスタファイル213への書き込みの競合を回避するようにするものである。

【0090】FAexch：FADD207のE1ステージにおいて、トラップ発生の可能性が否定できないとき、FADD207におけるパイプライン処理は通常のF1型からF2型になる。この場合、FADD207での実行終了ステージはMステージになるため、FADD207のE2及びE3ステージの間他のユニットの実行完了ステージを遅延させて全ての

10 ユニットがFADD207のMステージと同時に実行完了ステージに達するようにするために、このFAexchストール要求信号をアサートすることによって他のユニットの処理をストールする。このようなこのFAexchストール要求信号によってストールが起こる状況が、図13に示されており、ここでは「fadd」命令と同時にフェッチされた「ladd」「fmul」命令の処理と、次のサイクルでフェッチされる「fadd」「ladd」及び「fmul」命令の処理が「fadd」命令の処理のE2及びE3ステージ間にストールされる。

20 【0091】FMexch：FMUL208のE1ステージにおいて、トラップ発生の可能性が否定できない場合、FMUL208でのパイプライン処理は、通常のF1型からF2型タイプになる。この場合、FMUL208での実行完了ステージはMステージになるので、他のユニットの実行終了ステージを2サイクル遅延させて全てのユニットがFMUL208のMステージと同時に実行完了ステージに達するようにするために、このFMexchストール要求信号をアサートすることによって他のユニットの処理をストールする。

【0092】FDexch：FDIV209のE1ステージにおいて、トラップ発生の可能性が否定できない場合、FDIV209でのパイプライン処理は、通常のD1型からD2型になる。この場合、FDIV209での実行完了ステージはMステージになるので、他のユニットの実行完了ステージをFDIV209がE3ステージを通過する迄遅延させて全てのユニットがFDIV209のMステージと同時に実行完了ステージに達するようにするために、このFDexchストール要求信号をアサートすることによって他のユニットの処理をストールする。このようなこのFDexchストール要求信号によってストールが起こる場合が図14に示されており、「fdid」命令と同時にフェッチされた命令「iad」及び「fadd」の処理は、命令「fdid」がE3ステージを通過するまでストールされる。

【0093】Fstall：このタイプのストール要求信号は、実行完了ステージ以前にキャッシュミスリカバリのような処理を除く全てのパイプラインの処理をロックさせるために、外部からアサートされる。

【0094】図15から図17は、この第3実施例におけるストール制御のタイミングチャートの例を示す。

【0095】図15のタイミングチャートにおいては、50 N番目のサイクルでALU0 203のMステージにおいて

キャッシュミスが検出されて、M0busy信号がアサートされる一方で、キャッシュミスリカバリ処理がスタートする。一方、FMUL208についてトラップ発生の可能性が否定されないので、FMexch信号がアサートされ、stall11信号及びstall12信号がアサートされる。。

【0096】N+2番目のサイクルにおいては、ALU0203についてキャッシュミスリカバリ処理が続けられる一方、FMUL208は例外を生じなかったのでFMexch信号はニゲートされる。しかし、stall11信号及びstall12信号はまだアサートされているので、FMUL208の実行は完了できない。

【0097】N+4番目のサイクルにおいては、ALU0203についてキャッシュミスリカバリ処理がこのサイクル中に完了できるので、M0busy信号がニゲートされる。そのため、stall11信号及びstall12信号はニゲートされるので、他の全てのユニットでの実行が完了できるようになる。

【0098】そして最後に、N+5番目のサイクルにおいて、全ての命令の実行が完了されている。

【0099】図16のタイミングチャートにおいては、各処理ユニットでの命令のアドレスは、ALU0203<ALU1204且つFMUL208<ALU1204<FADD207であると仮定している。この場合、キャッシュミスがN番目のサイクルにおいてALU1204のMステージで検出されて、M1busy信号がアサートされる一方で、キャッシュミスリカバリ処理がスタートする。一方、トラップ発生の可能性がFMUL208について否定できないので、FMexch信号がアサートされ、stall11信号及びstall12信号がアサートされる。

【0100】N+2番目のサイクルにおいては、ALU1204についてキャッシュミスリカバリ処理が続けられる一方、FMUL208は例外を起こさなかったのでFMexch信号はニゲートされる。この時点で、stall12信号はニゲートされるのでALU0203及びFMUL208での実行は完了できるが、stall11信号がまだアサートされているので、FADD207の実行は完了できない。

【0101】N+4番目のサイクルにおいては、ALU1204についてキャッシュミスリカバリ処理がこのサイクル中に完了できるので、M1busy信号がニゲートされる。そのため、stall11信号はニゲートされるので、FADD207の実行が完了できるようになる。

【0102】そして最後に、N+5番目のサイクルにおいて、全ての命令の実行が完了されている。

【0103】図17のタイミングチャートにおいては、N番目のサイクルにおいてALU1204のMステージでキャッシュミスが検出されて、M1busy信号がアサートされる一方でキャッシュミスリカバリ処理がスタートする。一方、トラップ発生の可能性がFMUL208について否定できないので、FMexch信号がアサートされ、stall11信号及びstall12信号がアサートされる。

【0104】N+2番目のサイクルにおいては、ALU1204についてキャッシュミスリカバリ処理が完了し、M1busy信号がニゲートされ、同じN+2番目のサイクル又はひとつ前のN+1番目のサイクルにおいて、FMUL208が例外を起こさなかったのでFMexch信号がニゲートされる。この時点では、stall11信号及びstall12信号もニゲートされるので全ての命令の実行が完了できるようになる。

【0105】この第3実施例におけるstall11信号、stall12信号及びFRbusy信号を用いたストール制御処理に応じた各処理ステージでの各パイプラインの処理について図18に示される表に要約する。

【0106】この第3実施例においては、命令実行において例外発生の可能性を否定できないとして命令の処理がストールされた後、命令の処理は、例外が実際に命令実行の際に起こったときにアボートされるか、さもなくば、命令の実行において例外が実際には起こらなかったときに命令が復帰する。図18において、BUはプロセッサシステムの各処理ユニットに設けられた分岐ユニットを指すものである。

【0107】故に、この第3実施例によると、システムのクロック周波数の低下を防ぐように、サイクル時間を増加することなく処理可能なトラップとストールの制御機能を組み込んだスーパースカラプロセッサ等の並列処理型プロセッサシステムを提供することが可能となる。

#### 【0108】

【発明の効果】以上説明したように、本発明の並列処理計算機は、サイクル時間を増加させることなく、高速にトラップとストールの制御処理が可能なものであり、スーパースカラプロセッサ等の並列処理型プロセッサシステムにおいてシステムのクロック周波数を低下させることなくトラップとストールの制御をより効率的に行うことが可能となる。

#### 【図面の簡単な説明】

【図1】本発明に係る並列処理型プロセッサシステムの第1実施例のブロック図である。

【図2】図1の並列処理型プロセッサシステムにおけるトラップ制御ユニットのブロック図である。

【図3】図2のプログラムの実行時にトラップ要求が生じた場合の図1の並列処理型プロセッサシステムにおけるパイプライン処理の進行状況を示す図である。

【図4】本発明に係る並列処理型プロセッサシステムの第2実施例のブロック図である。

【図5】図4の並列処理型プロセッサシステムで実行されるプログラムの一例を示す図である。

【図6】図5のプログラムの実行時にトラップ要求が生じた場合の図4の並列処理型プロセッサシステムにおけるパイプライン処理の進行状況を示す図である。

【図7】本発明に係る並列処理型プロセッサシステムの第3実施例のブロック図である。

【図8】図7の並列処理型プロセッサシステムにおける、命令キャッシュミスによるストールの一例を示す図である。

【図9】図7の並列処理型プロセッサシステムにおける、命令キャッシュミスによるストールの別の例を示すタイミングチャートである。

【図10】図7の並列処理型プロセッサシステムにおける、命令キャッシュミスによるストールの別の例を示すタイミングチャートである。

【図11】図7の並列処理型プロセッサシステムにおける、命令キャッシュミスによるストールの別の例を示すタイミングチャートである。

【図12】図7の並列処理型プロセッサシステムにおける、命令キャッシュミスによるストールの別の例を示すタイミングチャートである。

【図13】図7の並列処理型プロセッサシステムにおける、FAexchによるストールの一例を示すタイミングチャートである。

【図14】図7の並列処理型プロセッサシステムにおける、FDexchによるストールの一例を示すタイミングチャートである。

【図15】図7の並列処理型プロセッサシステムにおける、ストール制御処理の一例を示すタイミングチャートである。

【図16】図7の並列処理型プロセッサシステムにおける、ストール制御処理の別の例を示すタイミングチャートである。

【図17】図7の並列処理型プロセッサシステムにおける、ストール制御処理の別の例を示すタイミングチャートである。

【図18】図7の並列処理型プロセッサシステムにおける、ストール制御処理に対する各パイプラインの処理様をまとめた表である。

【図19】従来のトラップ制御方法を用いた並列処理型プロセッサシステムのブロック図である。

【図20】図19の従来の並列処理型プロセッサシステムにおけるトラップ制御ユニットのブロック図である。

【図21】並列処理型プロセッサシステムで実行されるプログラムの一例を示す図である。

【図22】図21のプログラムの実行時にトラップ要求が生じた場合の図19の並列処理型プロセッサシステムにおけるパイプライン処理の進行状況を示す図である。

#### 【符号の説明】

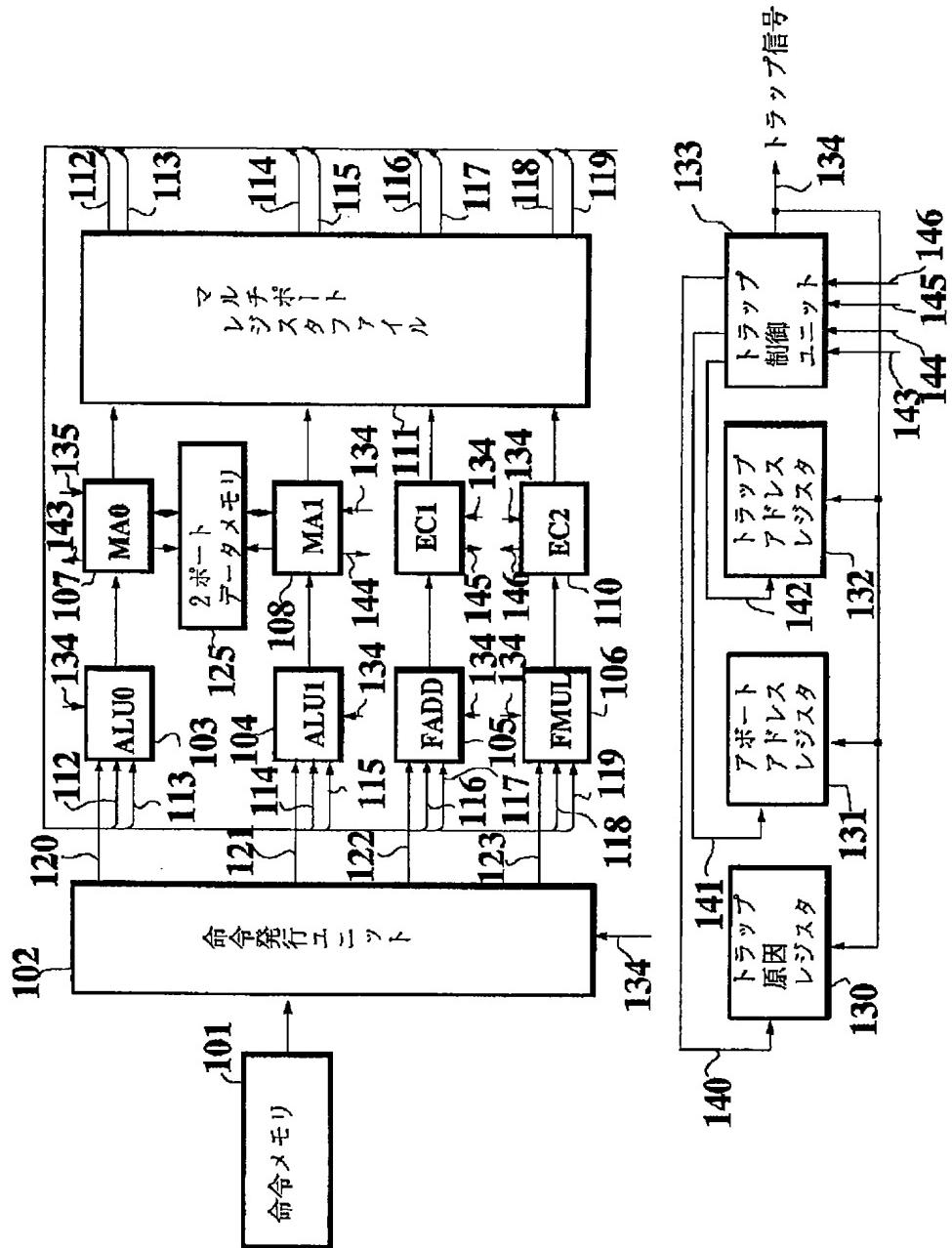
101 命令メモリ

101A 命令キャッシュメモリ

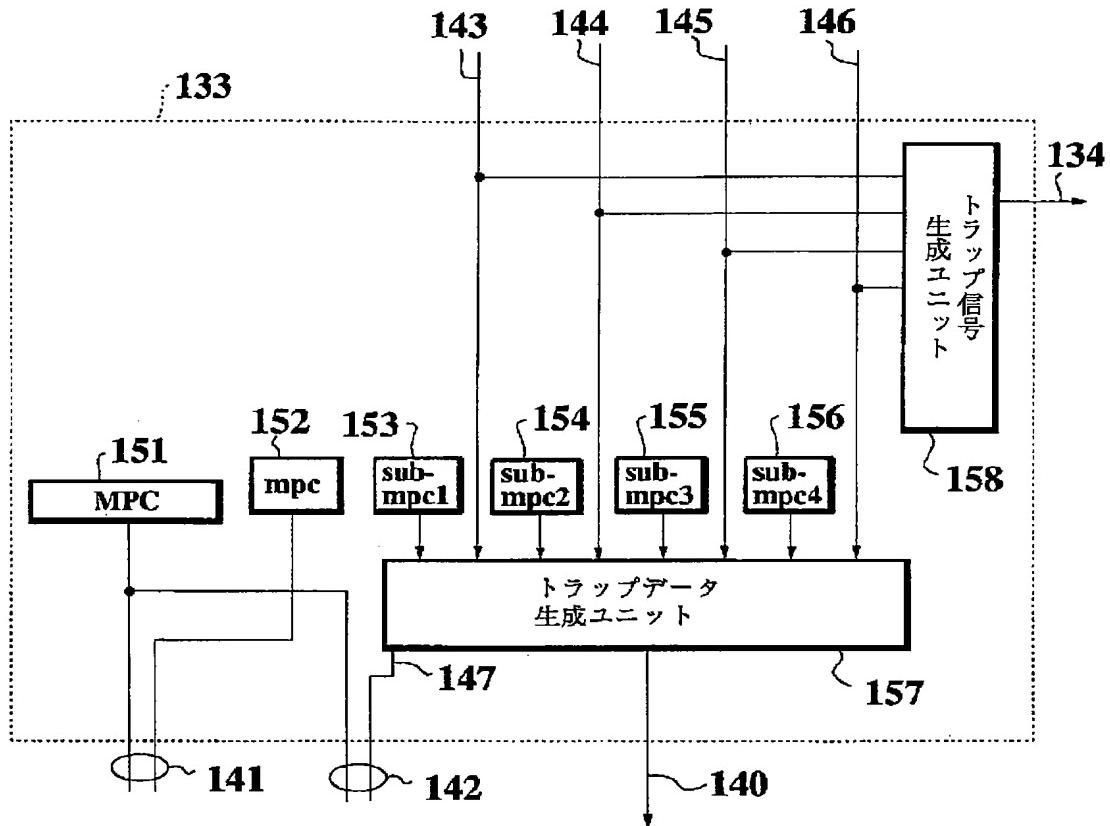
102 命令発行ユニット

- |      |                  |
|------|------------------|
| 102A | 命令発行ユニット         |
| 103  | 算術論理演算ユニット       |
| 104  | 算術論理演算ユニット       |
| 105  | 浮動小数点加算器         |
| 106  | 浮動小数点乗算器         |
| 107  | メモリアクセスユニット      |
| 108  | メモリアクセスユニット      |
| 109  | 浮動小数点例外チェックユニット  |
| 110  | 浮動小数点例外チェックユニット  |
| 111  | マルチポートレジスタファイル   |
| 125  | 2ポートデータメモリ       |
| 125A | 2ポートデータキャッシュメモリ  |
| 130  | トラップ原因レジスタ       |
| 131  | アボートアドレスレジスタ     |
| 132  | トラップアドレスレジスタ     |
| 133  | トラップ制御ユニット       |
| 151  | Mステージプログラムカウンタ1  |
| 152  | Mステージプログラムカウンタ2  |
| 153  | Mステージサブプログラムカウンタ |
| 154  | Mステージサブプログラムカウンタ |
| 155  | Mステージサブプログラムカウンタ |
| 156  | Mステージサブプログラムカウンタ |
| 157  | トラップデータ生成ユニット    |
| 158  | トラップ信号生成ユニット     |
| 160  | バスライン            |
| 161  | メインメモリ           |
| 162  | I/O装置            |
| 163  | ストール制御ユニット       |
| 201  | 命令キャッシュメモリ       |
| 202  | 命令発行ユニット         |
| 203  | 算術論理演算ユニット       |
| 204  | 算術論理演算ユニット       |
| 205  | 整数乗除算器           |
| 206  | 2ポートデータキャッシュメモリ  |
| 207  | 浮動小数点加算器         |
| 208  | 浮動小数点乗算器         |
| 209  | 浮動小数点除算器         |
| 210  | 命令アドレス生成ユニット     |
| 211  | 制御ユニット           |
| 212  | 整数レジスタファイル       |
| 213  | 浮動小数点レジスタファイル    |
| 214  | メインメモリ           |
| 215  | I/O装置            |
| 216  | バスライン            |
| 217  | プリデコーダ           |
| 218  | レジスタスコアボード回路     |

【図1】



【図2】



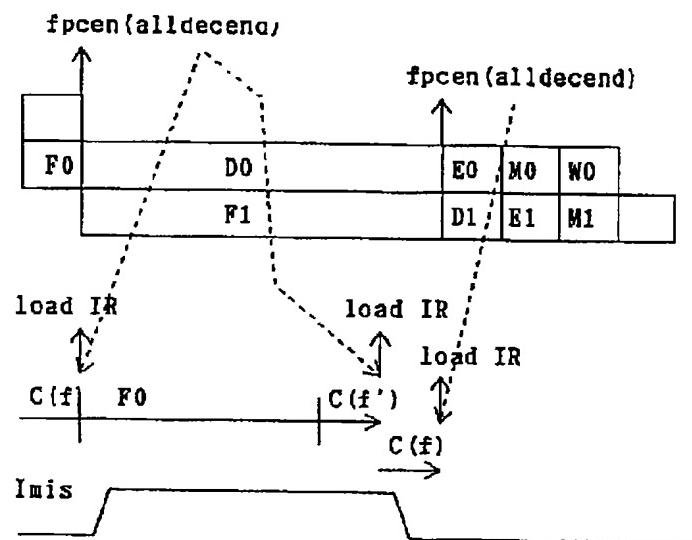
【図5】

```

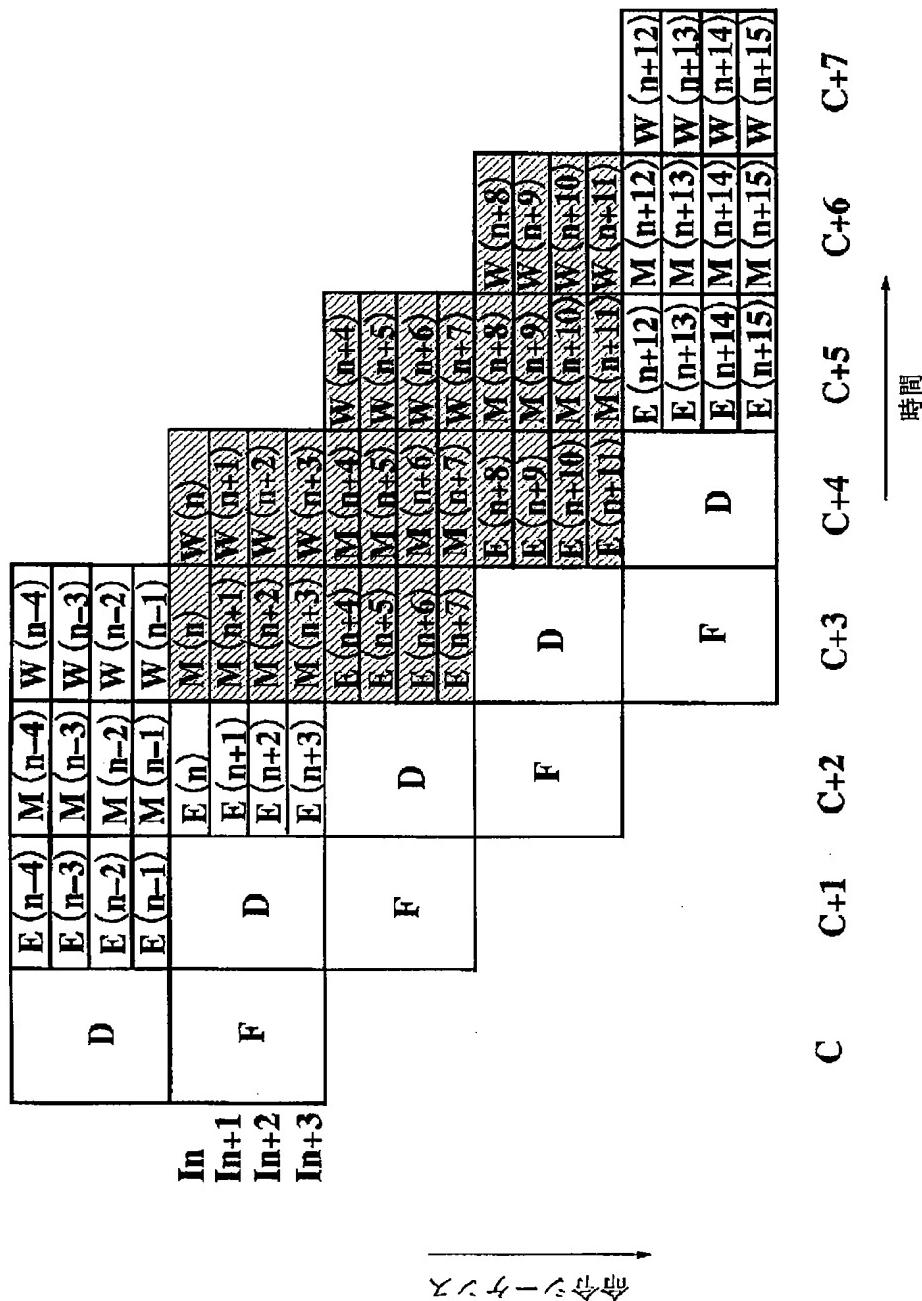
In-2
In-1
In   fadd fr1,fr2,fr3
In+1 load r1,r2,0×20
In+2 load r4,r1,0×30
In+3 fmul fr4,fr5,fr6
In+4

```

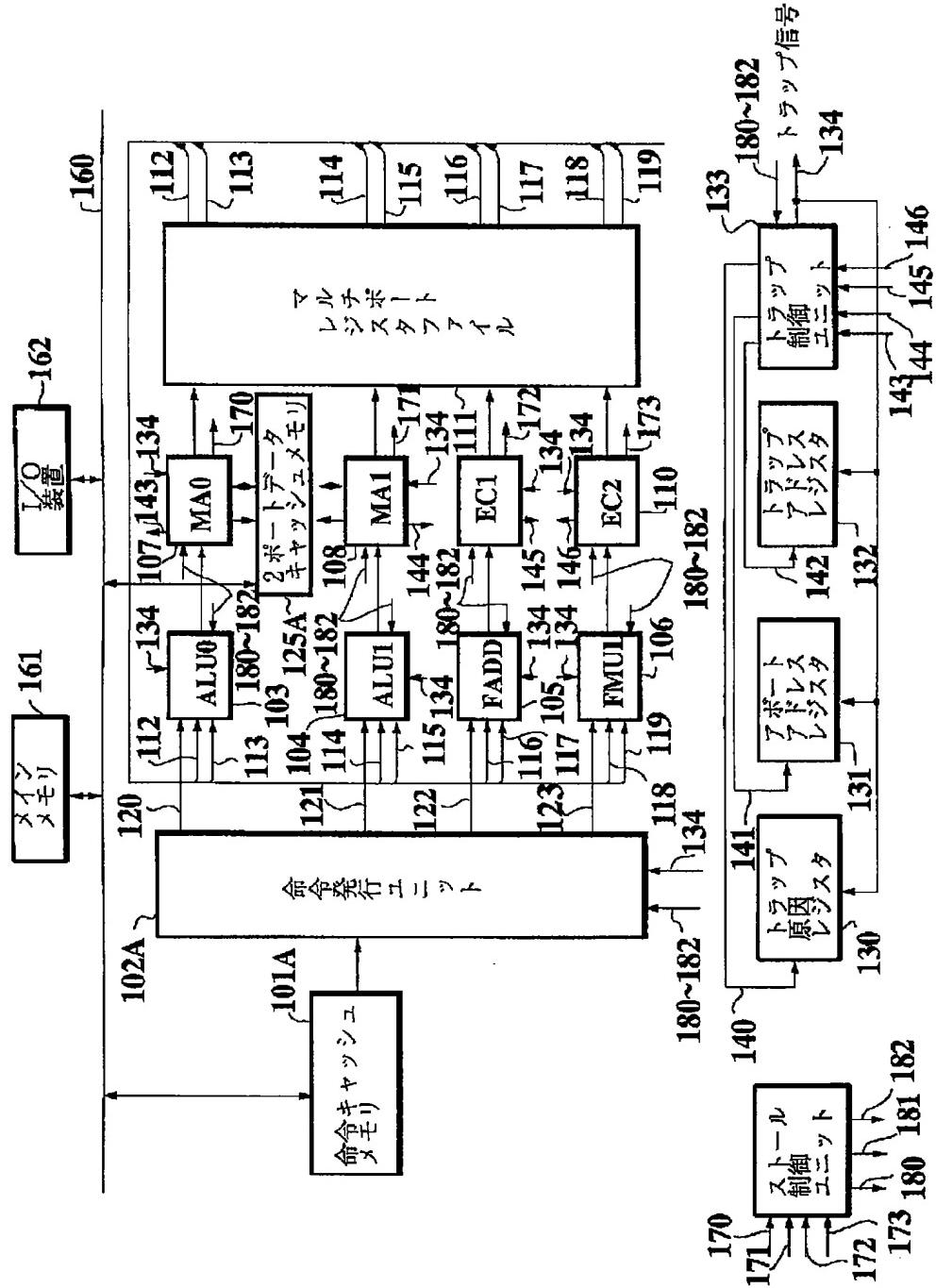
【図8】



【図3】



【図4】

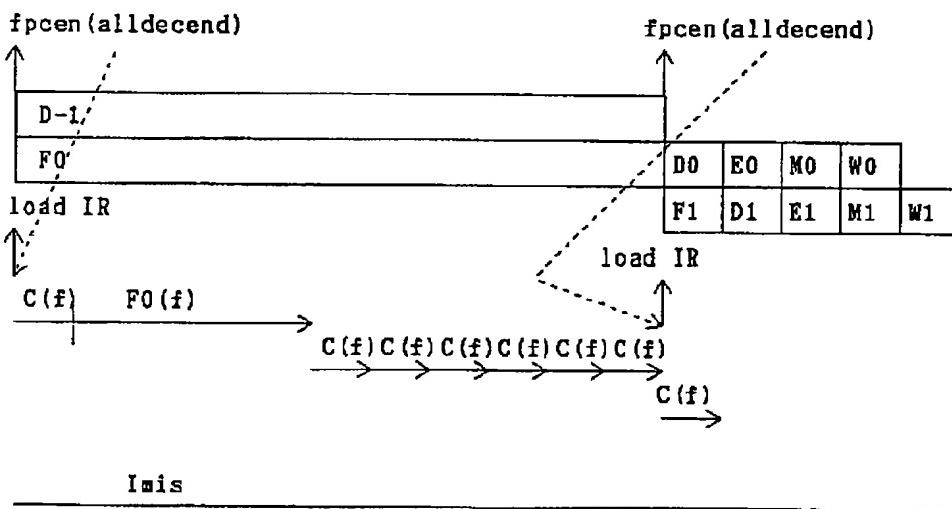


【図6】

$In$	$F_n$	$D_n$	$E(n)$	$M(n)$	$W(n)$	
$In+1$			$E(n+1)$	$M(n+1)$	$W(n+1)$	
$In+2$			$E(n+2)$	$M(n+2)$	$M(n+2)$	$W(n+2)$
$In+3$			$E(n+3)$	$M(n+3)$	$M(n+3)$	$W(n+3)$
	$F(n+4)$	$D(n+4)$	$E(n+4)$	$E(n+4)$	$M(n+4)$	$W(n+4)$
			$E(n+5)$	$E(n+5)$	$M(n+5)$	$W(n+5)$
			$E(n+6)$	$E(n+6)$	$M(n+6)$	$W(n+6)$
			$E(n+7)$	$E(n+7)$	$M(n+7)$	$W(n+7)$
	$F(n+8)$	$D(n+8)$	$E(n+8)$	$M(n+8)$	$W(n+8)$	
			$E(n+9)$	$M(n+9)$	$W(n+9)$	
			$E(n+10)$	$M(n+10)$	$W(n+10)$	
			$E(n+11)$	$M(n+11)$	$W(n+11)$	

$C \quad C+1 \quad C+2 \quad C+3 \quad C+4 \quad C+5 \quad C+6$

【図9】



【図14】

```

fdiv  F  D  E1 E1 E1 -----E1 E2 E3 M  W
      ↓   ↓   ↓   ↓   ↓   ↓ stall
iadd     E  M  M -----M  M  M  M  W
fadd     E1 E2 E2 -----E2 E2 E2 E3 W

```

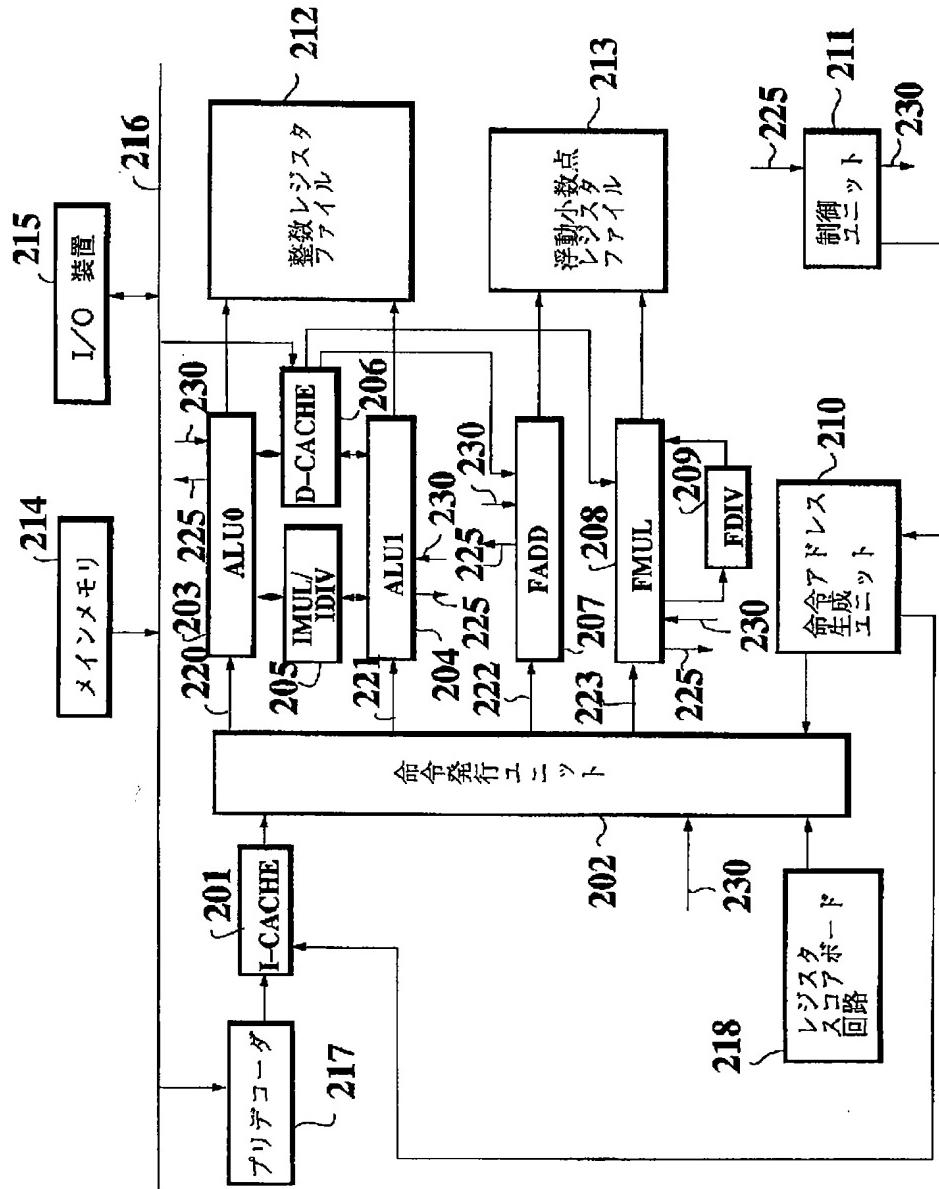
【図21】

```

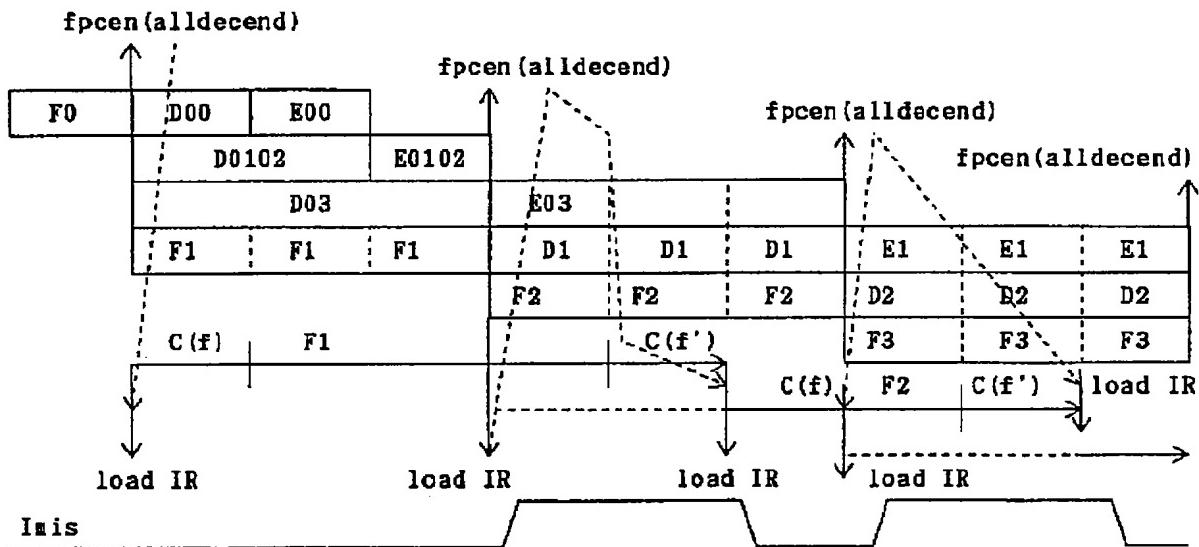
In-2
In-1
In fadd fr1,fr2,fr3
In+1 add r1,r2,r2
In+2 load r4,r1,O×10
In+3 fmul fr4,fr5,fr6
In+4

```

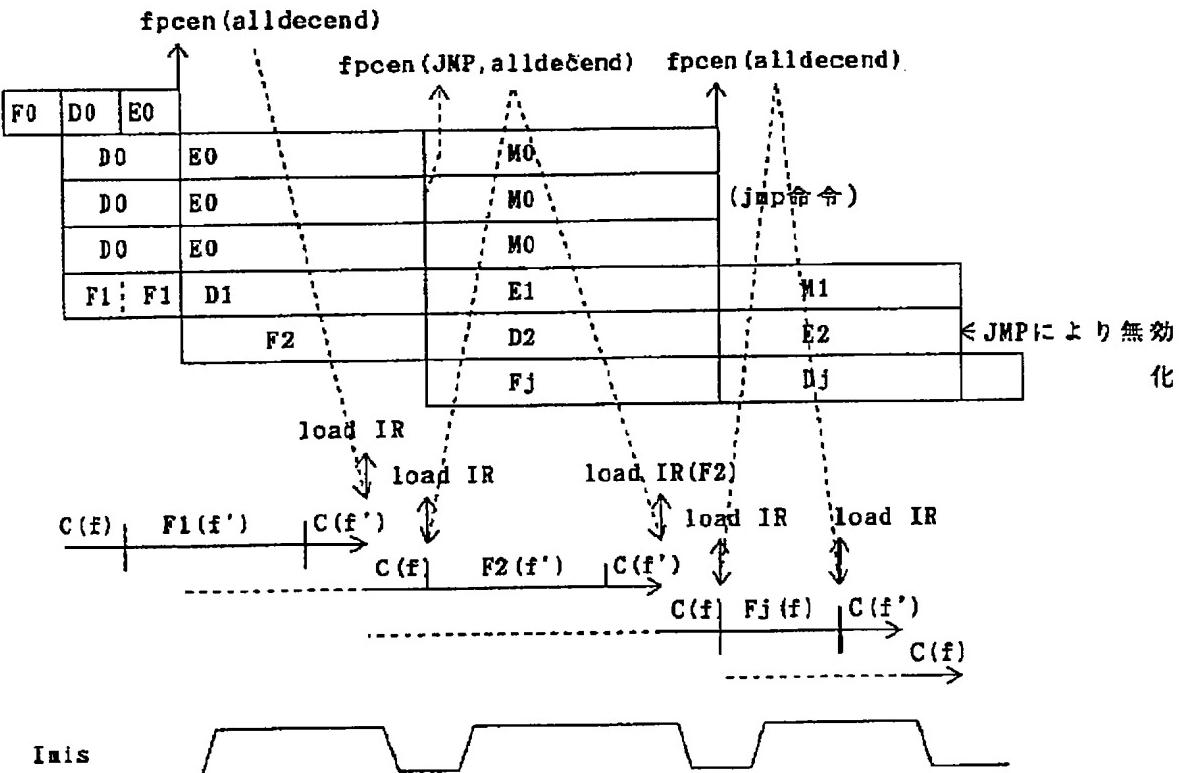
【図7】



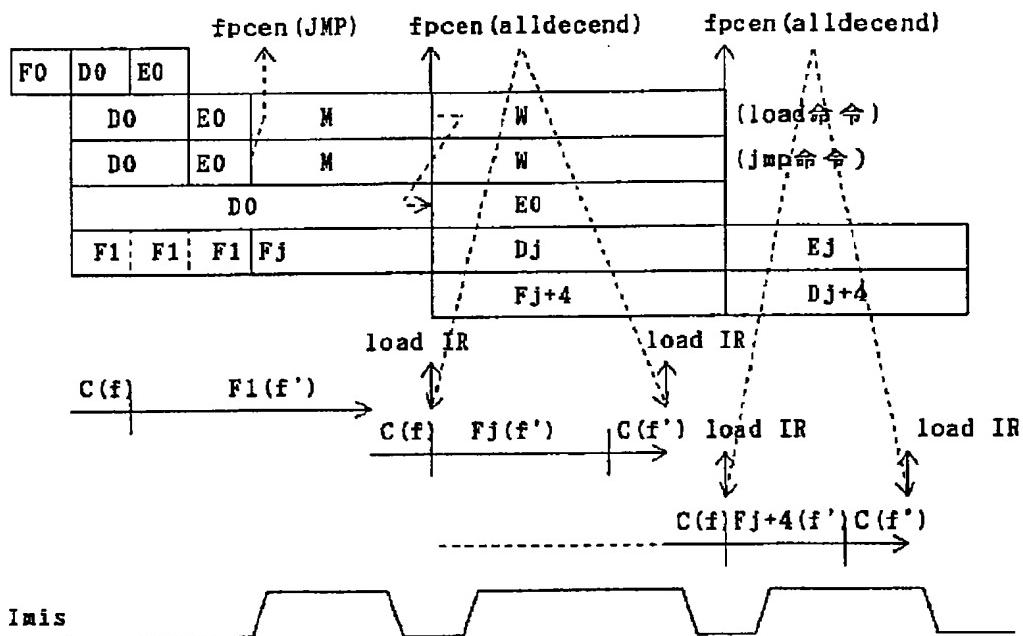
【図 10】



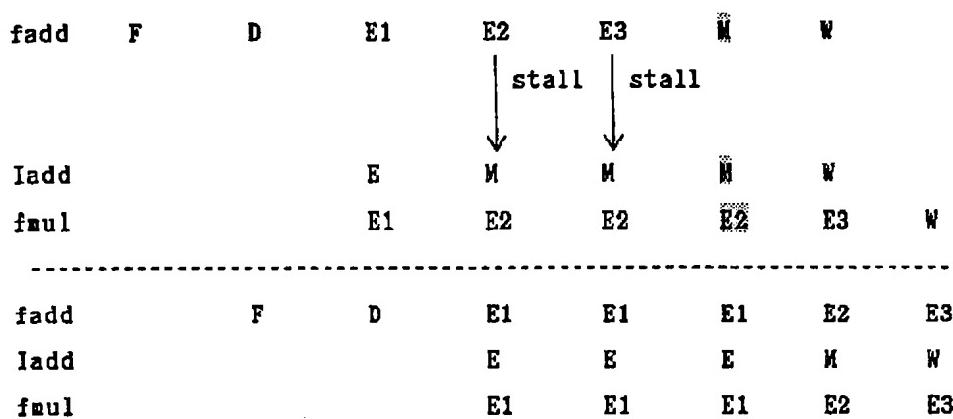
【图 1-1】



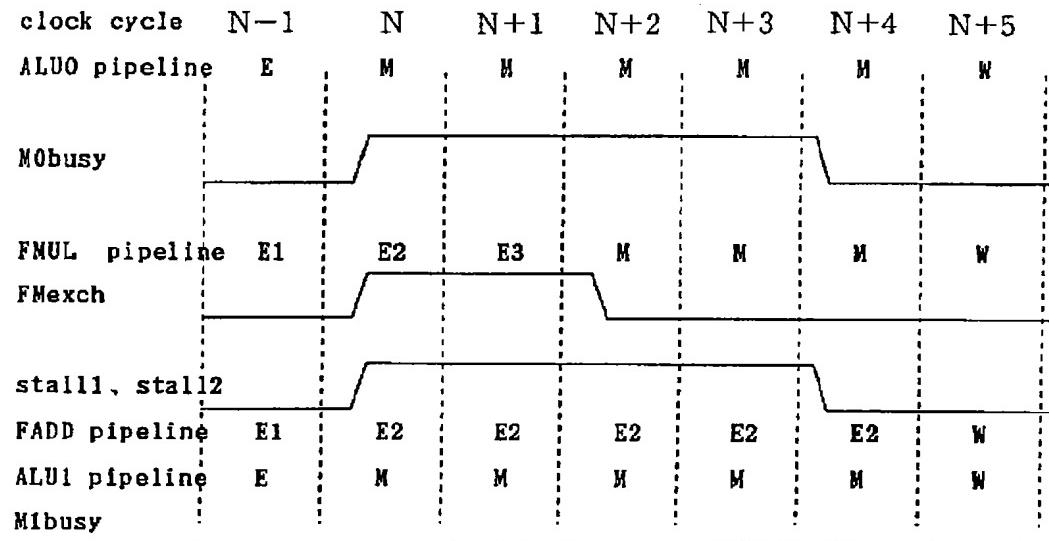
【図12】



【図13】

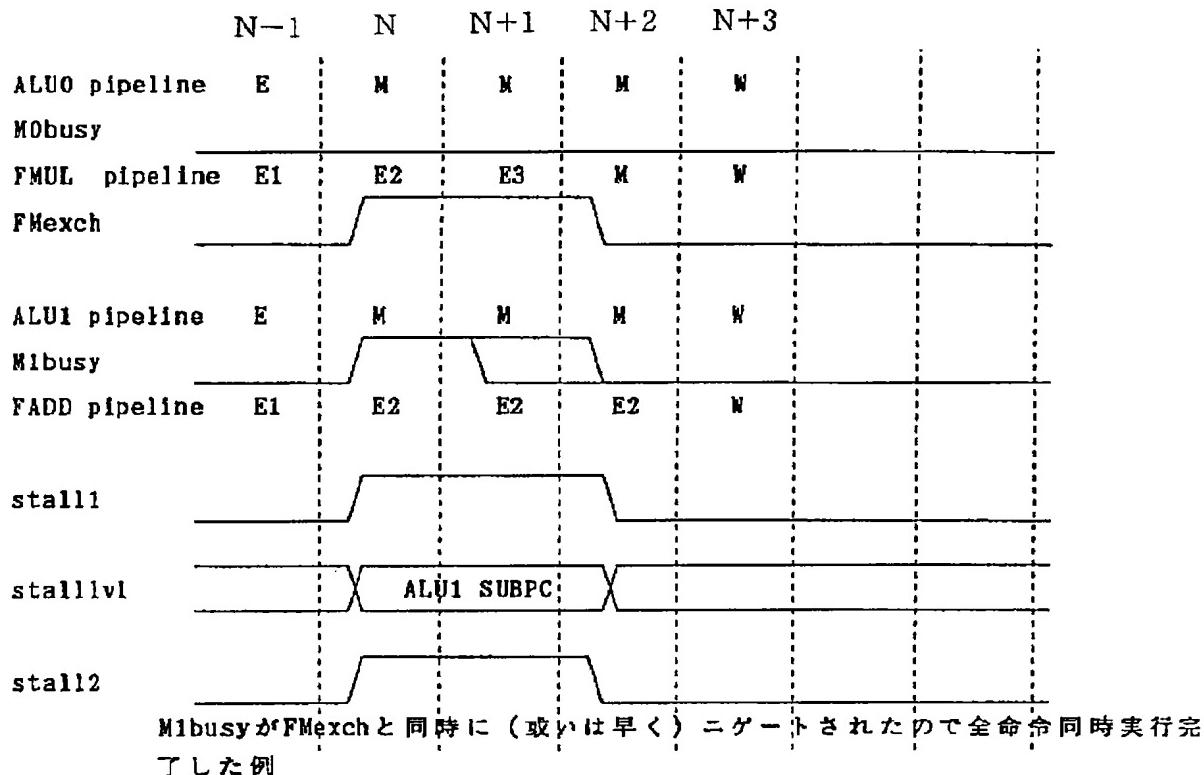


【図15】



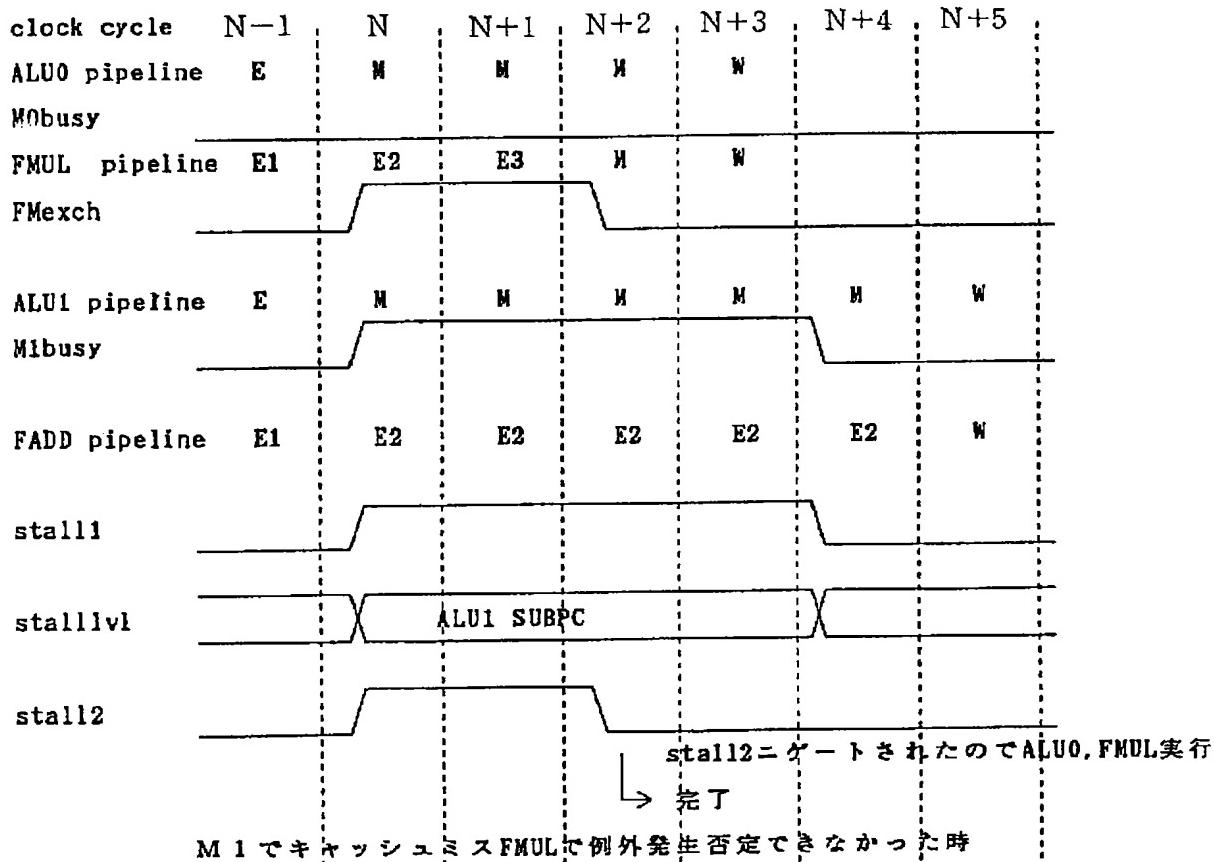
M0でキャッシュミスFMULで例外発生否定できなかった時

【図17】



【図16】

命令のアドレスは alu0<alu1, fmul<alu1<faddとする



【図18】

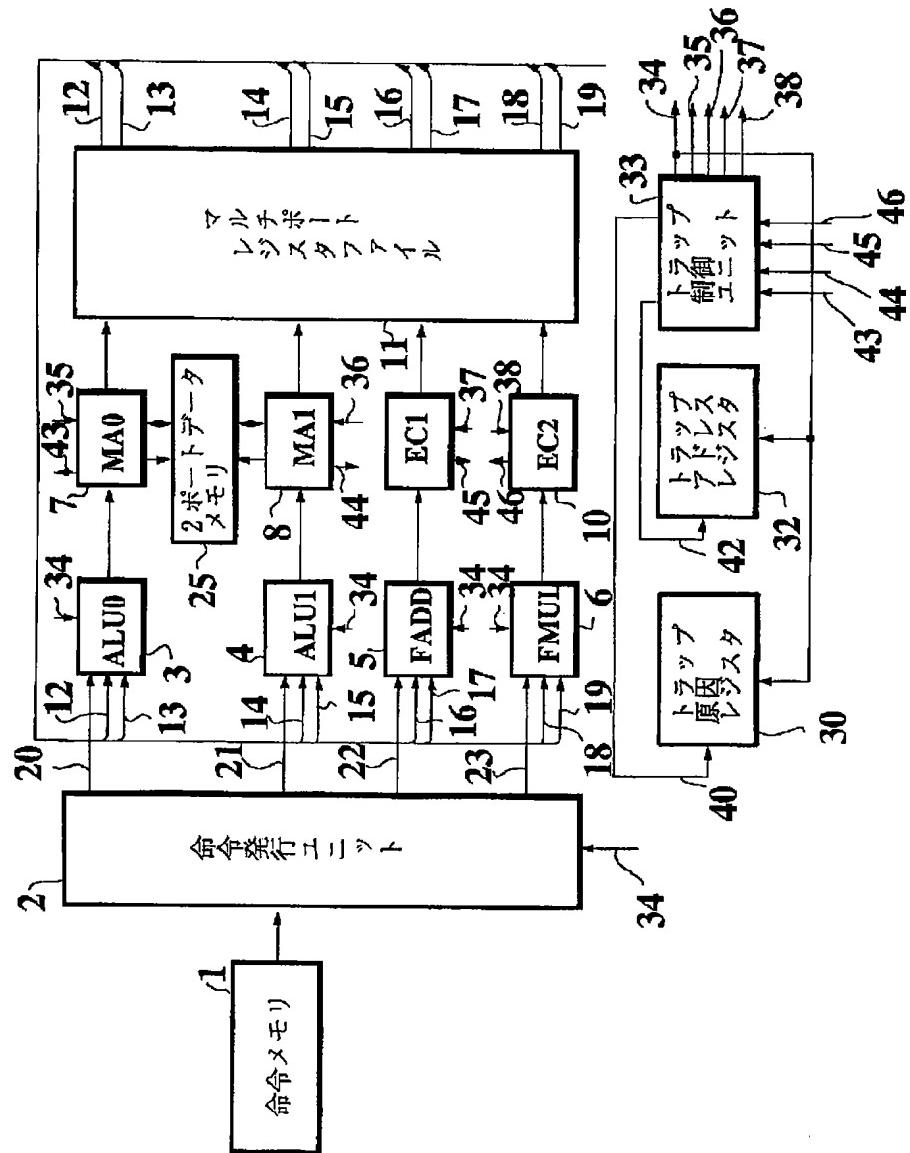
stall1	0	0	0	0	1	1	1	1	備考
stall2	0	0	1	1	0	0	1	1	
FRbusy	0	1	0	1	0	1	0	1	
F stage	○				×		×	×	
D stage	○				×		×	×	
ALU	E stage	○			×		×	×	
	M stage0	○			○		×	×	実行完了stage
	M stage1	○			×		×	×	実行完了stage
	M stage	○			○	有り得ない	○	○	
MUL/ DIV	E1 <sub>1</sub> stage	○			×	有り得ない	×	×	
	E1 <sub>2</sub> stage	○			△	有り得ない	×	×	実行完了stage
	E1 <sub>L</sub> stage	○			○	有り得ない	○	○	
	E1 <sub>w</sub> stage	○			○	有り得ない	○	○	
FADD (FMUL)	E1	○			×		×	×	
FAexch=	E2	○			△		×	×	実行完了stage
	E3	○			○		○	○	
	M stage	○			○		○	○	
FADD (FMUL)	E1 stage	○			×		×	×	
	E2 stage	○			○		○	○	
	E3 stage	○			○		○	○	
	ASSERT	M stage	○		△		×	×	実行完了stage
FDIV FAexch=	M stage	○			○		○	○	
	E1 <sub>1</sub> stage	○			×		×	×	
	E1 <sub>2</sub> stage	○			△		×	×	実行完了stage
	E1 <sub>L</sub> stage	○			○		○	○	
NEGATE	E2 stage	○			○		○	○	
	E3 stage	○			○		○	○	
	M stage	○			○		○	○	
FDIV FAexch=	E1 <sub>1</sub> stage	○			×		×	×	
	E1 <sub>2</sub> stage	○			○		○	○	
	E1 <sub>L</sub> stage	○			○		○	○	
	E2 stage	○			○		○	○	
ASSERT	E3 stage	○			○		○	○	
	M stage	○			△		×	×	実行完了stage
					○		○	○	
BU	E stage	○			×		×	×	
	M stage	○			△		×	×	
					○		○	○	

○ : ストールせず

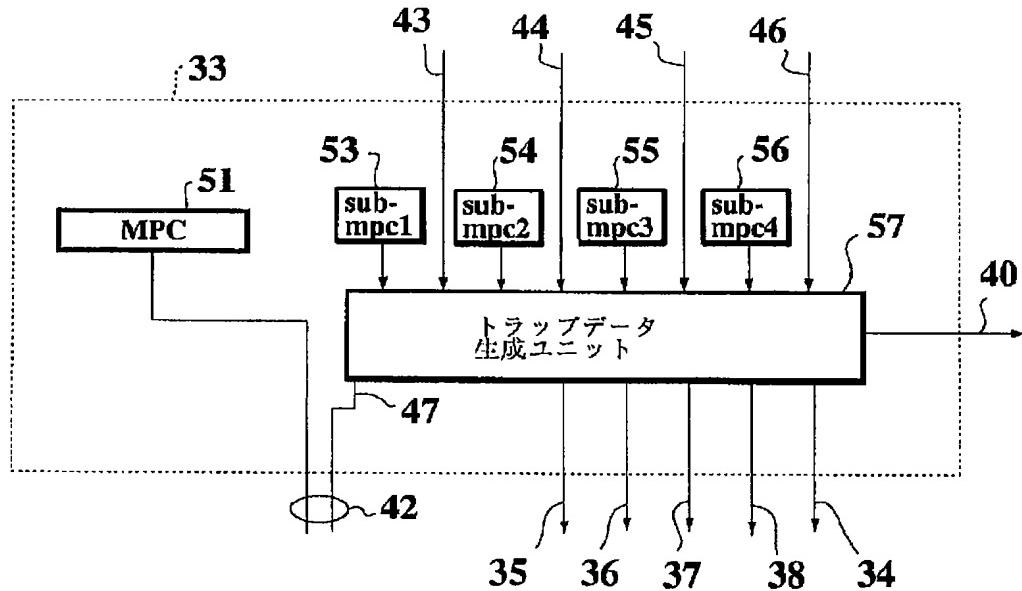
× : ストールする

△ : stall1 &gt; subpc ならストールせず、そうでなければストールする

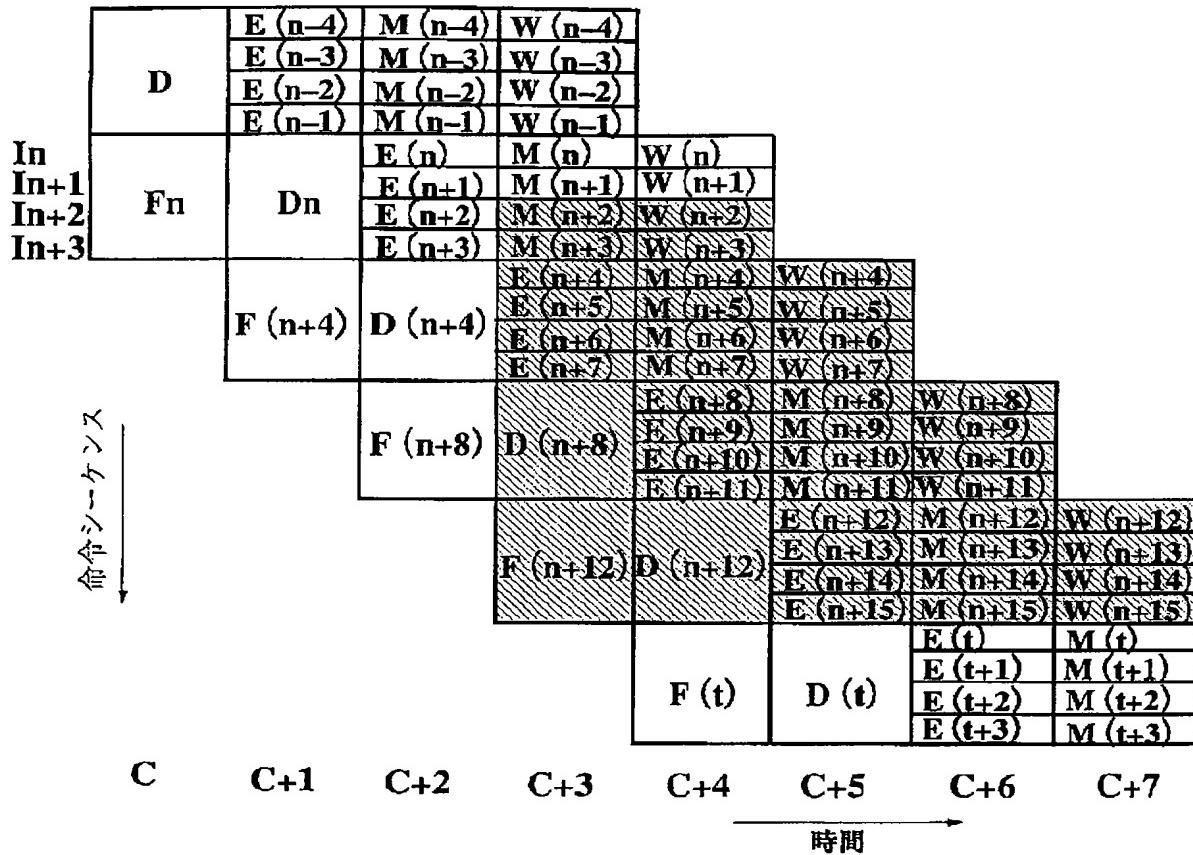
【図19】



【図20】



【図22】



フロントページの続き

(72)発明者 武田 譲治  
神奈川県川崎市幸区小向東芝町1 株式会  
社東芝総合研究所内

【公報種別】特許法第17条の2の規定による補正の掲載

【部門区分】第6部門第3区分

【発行日】平成11年(1999)11月30日

【公開番号】特開平5-181676

【公開日】平成5年(1993)7月23日

【年通号数】公開特許公報5-1817

【出願番号】特願平4-82490

【国際特許分類第6版】

G06F 9/38 380

310

370

11/00 310

15/16 390

【F I】

G06F 9/38 380 B

310 X

370 X

11/00 310 G

15/16 390 Z

### 【手続補正書】

【提出日】平成11年4月2日

### 【手続補正1】

【補正対象書類名】明細書

【補正対象項目名】特許請求の範囲

【補正方法】変更

【補正内容】

### 【特許請求の範囲】

【請求項1】 複数シーケンスの命令を同時実行するN個(Nは正整数)の命令実行手段と、前記N個の命令実行手段により同時実行される命令を供給する命令供給手段と、M個(N≥M、Mは正整数)の命令が前記命令供給手段から前記N個の命令実行手段に同時に供給され、一クロックサイクル中に前記M個の命令の中の少なくとも一つの命令実行において処理例外が発生したとき、該一クロックサイクル中に前記N個の命令実行手段の全てにアボート信号を送ることにより前記N個の命令実行手段に供給された前記M個の命令の同時処理を全て該一クロックサイクル中にアボートするように前記N個の命令実行手段を制御するトラップ制御手段と、を備えたことを特徴とする命令レベル並列処理型プロセッサシステム。

【請求項2】 前記トラップ制御手段により処理がアボートされた前記M個の命令の中で最も小さいアドレスを有する命令のアドレスを格納するアボートアドレス格納手段と、

前記M個の命令の中で前記処理例外を発生させた命令のアドレスを格納するトラップアドレス格納手段と、

を更に備えたことを特徴とする請求項1記載の命令レベル並列処理型プロセッサシステム。

【請求項3】 前記N個の命令実行手段は、K個(M≥K、Kは整数)の同等の機能を有する順序付けされた演算手段を有し、前記命令供給手段はJ個(K≥J>1、Jは整数)の順序付けされた命令を、該J個の命令の中でより前の順序にある命令が該K個の演算手段の中でより前の順序にある演算手段に供給されるように、該K個の演算手段に供給することを特徴とする請求項1記載の命令レベル並列処理型プロセッサシステム。

【請求項4】 前記J個の命令中のI番目の命令の実行において処理例外が発生する可能性を否定しきれないときには、前記J個の命令の中で該I番目の命令よりも後の順序にある命令の処理をストールするように、前記J個の命令の一部の処理をストールするストール手段を、更に備えたことを特徴とする請求項3記載の命令レベル並列処理型プロセッサシステム。

【請求項5】 前記M個の命令の実行において処理例外が発生する可能性を否定しきれないときには、前記M個の命令の処理をストールし、前記M個の命令の実行において処理例外が実際に発生したときには、前記M個の命令の処理をアボートし、前記M個の命令の実行において処理例外が実際に発生しなかったときには、前記M個の命令の処理を再開するストール制御手段を、更に備えたことを特徴とする請求項1記載の命令レベル並列処理型プロセッサシステム。

【請求項6】 前記処理例外に対処する手段と、前記処理例外に対処した後に、前記トラップ制御手段に

よりアボートされた前記M個の命令を同時に再スタートする手段と、  
を更に備えたことを特徴とする請求項1記載の命令レベル並列処理型プロセッサシステム。

【請求項7】 命令レベル並列処理型プロセッサシステムの制御方法であって、  
該システムのN個（Nは正整数）の命令実行手段により  
同時実行される複数シーケンスのM個（N≥M、Mは正  
整数）の命令を供給するステップと、  
前記M個の命令が前記N個の命令実行手段に同時に供給

され、一クロックサイクル中に前記M個の命令の中の少  
なくとも一つの命令実行において処理例外が発生したと  
き、該一クロックサイクル中に前記N個の命令実行手段  
の全てにアボート信号を送ることにより前記N個の命令  
実行手段に同時に供給された前記M個の命令の処理を全  
て該一クロックサイクル中にアボートするように前記N  
個の命令実行手段を制御するステップと、  
を備えたことを特徴とする命令レベル並列処理型プロセ  
ッサシステムの制御方法。